93-9

93-9

LEARNING TO RECOGNIZE VISUAL CONCEPTS:

Development and Implementation of a Method for Texture Concept Acquisition Through Inductive Learning

Jerzy W. Bala

MLI ~~93-3~~
93

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
1993		**00-00-1993 to 00-00-1993**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Learning To Recognize Visual Concepts: Development and Implementation of a Method for Texture Concept Acquisition Through Inductive Learning	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **George Mason University,4400 University Drive,Fairfax,VA,22030**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT
see report

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**129**	

Learning To Recognize Visual Concepts: Development and Implementation of a Method for Texture Concept Acquisition Through Inductive Learning

A dissertation submitted in partial fulfillment of the requirement for the degree of Doctor of Philosophy at George Mason University.

By

Jerzy Wojciech Bala
Master of Science
Electrical and Computer Engineering
AGH Polytechnic University
May 1985

Director: Dr. Ryszard Spencer Michalski
Professor, Computer Science Department, and
Systems Engineering Department

Spring 1993
George Mason University
Fairfax, Virginia

DEDICATION

To My Wife Alicja Maria and
My Parents, Helena and Andrzej

ACKNOWLEDGMENTS

I would like to first of all thank my advisor, Dr. Ryszard Michalski, who introduced me to the field of Artificial Intelligence, for his advice, support and encouragement throughout my dissertation research. The approaches developed in this thesis are based on his ideas of utilizing machine learning techniques in computer vision.

I would also like to thank my committee members Dr. Kenneth DeJong, Dr. Kenneth Hintz, and Dr. Peter Pachowicz, for their valuable help, insightful comments, recommendations, and suggestions to this research. They have influenced this work and encouraged me to think more deeply about the research.

I appreciate the friendship and the working environment provided by the members of the Machine Learning and Inference Laboratory. My fellow graduate students, Eric Bloedorn, Janusz Wnek, Mike Hieb, Tom Dybala, Ibrahim Imam, Haleh Vafaie, and Ken Kaufman, not only gave useful advice but also were good friends. Thanks go to Eric Bloedorn and Mark Maloof for their patient proofreading of the manuscript.

Closer to home, I am most grateful to my parents for their encouragement, and faith in me throughout the years. The selfless support provided by them is unforgettable. Finally, I would like to express my gratitude to my wife, Alicja Maria, for her love and patience. She supported my dissertation research well beyond the call of duty. I hope to now make up for the long hours I have been at work over the last years.

My one semester stay at the Computer Science Department of the University of Illinois at Urbana-Champaign was supported by the grant form the Kosciuszko Foundation. I am grateful for their support.

This research has been done in the George Mason University Center for Artificial Intelligence and has been supported in part by the Defense Advanced Research Projects Agency under the grant administered by the Office of Naval Research No. N00014-87-K-0874 and No. N00014-91-J-1854, in part by the Office of Naval Research under grants No. N00014-88-K-0397, No. N00014-88-K-0226 and No. N00014-91-J-1351, and in part by the National Science Foundation Grant No. IRI-9020266. I am grateful to these agencies for their support.

TABLE OF CONTENTS

LIST TABLES

LIST OF FIGURES

ABSTRACT

LEARNING TO RECOGNIZE VISUAL CONCEPTS:
DEVELOPMENT AND IMPLEMENTATION OF A METHOD
FOR TEXTURE CONCEPTS ACQUISITION
THROUGH INDUCTIVE LEARNING

JERZY WOJCIECH BALA, Ph.D.
George Mason University, May 1993
Dissertation Director: Dr. Ryszard S. Michalski

The goal of this research is to explore the application of symbolic learning methods to problems of computer vision. The research presented in this thesis has been concerned primarily with the development of methods for inductive learning of texture descriptions. Texture description learning is done in the following phases: (i) data pre-processing and attribute extraction, (ii) acquisition of texture concept descriptions, (iii) optimization of acquired descriptions, and (iv) recognition of unknown texture samples. The methodology adapted to the acquisition and recognition of complex vision data is based on an extension of *AQ* [Michalski, 1986], a learning from-examples algorithm. This approach for inductive learning of texture descriptions was originally proposed by Michalski [1973] and was initially applied using ILLIAC III image recognition computer facilities. This research presents a novel extension to the initial approach, which is called *Multilevel Logical Templates*. The novelty lies in multilevel symbolic image transformations, new advanced concept description optimization methods for noise-tolerant learning, and a multistrategy approach to learning from vision data. An important contribution of the research is the experimental demonstration that symbolic inductive learning methods can be successfully applied to the domain of continuous attributes of low level vision in which non symbolic methods have been traditionally employed.

Chapter 1

1. Introduction

Intelligent behavior is characterized by the ability to learn and to self-organize. Such abilities include self-creation of knowledge through experience, self-improvement of control through practice, discovery of new concepts and relationships through observation and analogy, and reorganization of knowledge to perform a given task better. Considering human or animal vision, it is clear that the ability to recognize and handle visual objects to a large extent is accomplished through learning, and only partially by genetic pre-programming. Thus, we should be aware of the need for learning in enhancing computational vision capabilities. However, relatively little effort has gone into actually using learning in vision and much of past learning in vision was restricted to statistical methods. Only until recently has there been some increased interest in learning in vision, mainly in the form of the renewed interest in neural networks. Even in the early 1980's, there were no learning paradigms in computational vision. As cited in Ballard and Brown's *Computer Vision* book [Ballard and Brown, 1982];

> *Learning is missing from the list above. Disappointing as it is, at this writing the problem of learning is so difficult that we say very little about it in the domain of vision .* *

Further progress in computer vision depends on the implementation of various learning strategies in vision systems. The reasons for this view are based on the following observations [Michalski, Bala et al., 1993]:

---

* page 315, a paragraph describing the organization and topics of chapter IV in Ballard's book.

- *The world changes in unpredictable ways, therefore it is impossible, in principle, to pre-program in the vision systems all the knowledge necessary for image understanding.*

- *Handcrafting the knowledge needed for image understanding into computer vision systems is a difficult and time-consuming process; learning provides a fundamental vehicle for simplifying this process.*

- *In biological vision systems, many aspects of image perception are genetically preprogrammed, but many are learned. Similarly, computer vision systems should be able to acquire some capabilities through learning.*

The goal of this research is to explore the application of machine learning methods to problems of computer vision. The research presented in this thesis has been concerned primarily with the development of methods for inductive learning of texture descriptions. Learning of texture description is separated into the following phases:

(i) image pre-processing (volume optimization),
(ii) attribute extraction,
(iii) acquisition of texture concepts by inductive learning,
(iv) optimization of concept prototypes, and
(v) recognition of unknown texture samples.

The texture description acquisition method, adapted to the acquisition and recognition of complex vision data, is based on an extension of AQ, a learning from-examples algorithm. The approach for inductive learning of texture descriptions was originally proposed by Michalski [Michalski, 1973] and was initially applied using ILLIAC III image recognition computer facilities. This research presents a novel extension to the initial approach. It uses logic-style rules, called Multilevel Logical Templates. The novelty also lies in new advanced concept description optimization methods for noise-tolerant learning, and a multistrategy approach to learning from vision data. The following methods (and implemented systems) are described in the thesis:

TEXTRAL: the primary method for learning texture descriptions described in this thesis. TEXTRAL employs multilevel symbolic image transformations called Multilevel Logical Templates (MTL) and the AQ15 inductive learning program.

AQ-NT: a method for learning of reduced complexity rule sets, from noisy inputs, and

AQ-GA : a method combining inductive rule learning with a genetic algorithm for rule enhancement.

An important result of this research is a demonstration that symbolic learning methods can be successfully applied to selected problems of low-level vision, in which nonsymbolic methods have been traditionally employed. Specifically, the results obtained demonstrate that these methods have been very useful for creating descriptions of textures from their samples, obtained from the original camera-generated images.

The texture data is the initial domain of experiments reported in this dissertation research. However, the presented methods can be applied to learning in any domain characterized by continuous attributes, noisy data, multiclass environments, and complex representation spaces. Characteristics of such a domain are apparent for engineering data. Learning from engineering data require new learning tools. that are noise-tolerant, capable of processing complex data including multiclass environments, and large training sets. These tools are expected to significantly extend the state of the art and open up a whole new application area for machine learning.

1.1 Background

1.1.1 Inductive Learning

The problem investigated in this thesis falls into the category of inductive learning of visual concepts from examples. In the learning from examples paradigm, a set of training examples annotated with concept membership information is used as the basis for

automatically inducing a general description for each visual concept. The concept description learned is correct for the given examples. Since it extends its membership information to unseen parts of the representation space it is also a good predictor for the classification of unobserved examples of the concept.

An example in this paradigm may be anything that can be expressed in terms of representation language. An example can be a physical object, a situation, a cause, or a concept. Training examples are usually described in one of the two types of representation languages: attribute-based or predicate-based. In an attribute-based representation, an example is represented as an n-tuple of attributes values, where n is a number of attributes. All n attributes define the event space. A domain is associated with each attribute used to describe examples. The domain indicates the values the attribute may assume. The values in a domain may be unordered (or nominal), ordered (or linear), or hierarchically structured [Michalski, 1983]. A predicate-based representation allows examples to be represented as structural descriptions. In structural descriptions, each example may consist of several objects, and a set of relationships among these objects. A predicate-based representation is more powerful than an attribute-based representation, but the limited expressiveness of an attribute-based representation allows relatively efficient learning algorithms to be designed. The attribute-based representation can be used in many real world applications.

Concept descriptions may be described in either of the two representation languages. In this thesis, an attribute-based representation is selected to describe both examples and concepts of texture classes.

Most inductive learning systems generate concept descriptions by detecting and describing similarities among positive examples and dissimilarities between positive and negative examples. Inductively constructing concept descriptions from training examples involves the transformation of training examples using a set of refinement operators [Michalski, 1983]. A refinement operator is either a specialization or a generalization operator. When applied to hypothesis or a training example, a generalization/specialization operator transforms it into a more general/special hypothesis.

Each hypothesis describes a subset of all examples, while all hypotheses representable in a given representation language from a hypothesis space. Learning can be viewed as a search process through the hypothesis space to find description of the target concept. Generalization/specialization operators are search operators. Search heuristics are some preference criteria (also called biases). One of the most important description preference criterion is accuracy. Hypothesis accuracy depends on the completeness and consistency of this hypothesis with regard to the learning examples. Simplicity and comprehensibility are two other preference criteria.

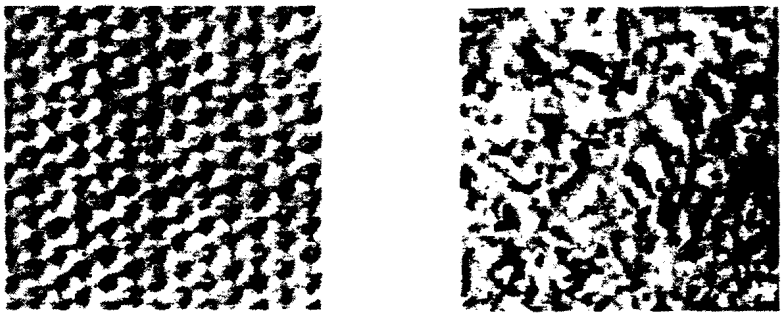
An inductively generated description should not only classify training examples, but also unseen examples, so it should be more general than training examples. Unfortunately, induction is an error prone process. Induction is falsity preserving rather than truth preserving. That is, a concept description inductively generated from examples cannot be guaranteed correct. It may be only an approximation of the concept.

Concept descriptions produced by early inductive learning algorithms, however, can be complete and consistent with respect to the training examples. A description is consistent if it covers all training examples of the concept (positive examples), while a description is complete if it does not cover any counter examples of the concept (negative examples). In order to achieve completeness and consistency in the presence of noise, a complex description can be generated. This is the well known phenomenon of overfitting. The prediction accuracy of a complex description for unseen examples may be inferior to a simpler, less complex description. For example, a fully expanded decision tree or highly disjunctive rule may cover training examples completely. However a smaller tree or rule set, with a larger apparent error on the training examples, may be more accurate in its predictions for unseen examples. Description complexity and prediction accuracy are highly related. Many newly developed approaches, such as AQ16 [Zhang and Michalski, 1989], C4 [Quinlan, 1987], CN2 [Clark and Niblett, 1989], and PLS1 [Rendell, 1983], allow the descriptions produced to be incomplete and/or inconsistent with respect to the training data. The issue of learning reduced complexity rule sets in learning from complex texture data is addressed in this dissertation research.

1.1.2 The Texture Analysis Problem

Texture can be described in a variety of ways. It can be generated by primitives organized by placement rules or as the result of some random process. It can be found in a continuous spectrum from purely deterministic to purely stochastic. The different textures in an image are usually very apparent to a human observer, but an automatic description of these patterns has proven to be complex.

Visual textures arise from many sources. Cellular textures are composed of repeated similar elements called primitives. Examples are leaves on a tree or bricks in a wall. Other texture types include flow patterns, fiber masses, and stress cracking. Texture can be both structured and random (Figure 1-1). It is common to speak of a uniform texture or a homogeneous texture, despite the apparent contradiction. This homogeneity is a perceptual phenomenon. Somehow the human visual system analyzes images and measure texture properties. Some texture fields are seen to be equivalent, others to differ in coarseness, linearity, or other texture dimensions. All, however, are unified by their perception as texture fields.



(a) a structural texture (b) a random texture

Figure 1-1: Examples of textures

Texture provides very useful information for the automatic interpretation and recognition of the image by a computer. Textural features can be crucial for the segmentation of an image and can serve as the basis for classifying image parts. Many, if not all, objects in one's familiar environment can be recognized on the basis of just these two properties (i.e., without information about their shape, size or other characteristics).

Traditionally, all methods of textural analysis have taken either the statistical approach, in which the statistical properties of the spatial distributions of the gray levels are used as the texture descriptors, or the structured approach which conceives texture as an arrangement of a set of spatial subpatterns according to a certain placement rules.

The structural texture models are best suited to situations in which complete descriptions of individual texture primitives are derivable from the image. This usually means that the texture primitives consist of relatively large numbers of pixels, and that the boundaries of the primitives are consistently discernible. The statistical model usually describes texture by statistical rules governing the distribution and relation of gray levels. This class of models involves the use of statistical tools. The statistical texture models are suitable when the sizes of the texture primitives tend to be on the order of few pixels. The statistical approach works well for many textures which have barely discernible primitives. However, it can also be effective in cases of large texture primitives if the boundaries of the primitives are highly convoluted, or if the interior areas are not completely homogeneous in intensity. A disadvantage of this statistical method is that it is highly dependent on the chosen resolution.

The dichotomy between these two classes, however, is not clear-cut, since statistical tools and concepts are introduced into models which are basically structural, and statistical models can describe pattern-like textures and vice versa. This division is therefore sometimes artificial.

Different statistical and structural techniques have been developed. These techniques include, for example: co-occurrence matrices (e.g., [Haralick, Shanmugan et al., 1973, Davis, Clearman et al., 1981]), coarseness measures (e.g., [Zucker, Rosenfeld et al., 1975]), statistical measures (e.g., [Haralick, 1979, Pratt, Faugeras et

al., 1978]), texture filters and energy measures (e.g., [Laws, 1980]), Fourier transform (e.g., [Bajcsy and Lieberman, 1976]), Markov random fields [Cross and Jain, 1983], Gibbs random fields [Derin and Elliott, 1987], and many others. Texture macro-structural properties were also modeled incorporating shape and tree grammars [Rosenfeld and Lipkin, 1970, Lu and Fu, 1978].

Recently model-based approaches have been used for texture characterization. Model-based approaches attempt to capture the dependencies among neighboring pixel values by fitting an analytical function (model) to the texture image. Most model-based techniques treat texture as a realization of a two-dimensional stochastic process, or random field. Once an appropriate model of a given texture has been found, the parameters of the model would completely specify the texture. Some of the well-known model-based techniques for texture classification and segmentation are based on Markov random field (MRF) models [Besag, 1986, Chellappa, 1985], mosaic models [Ahuja and Rosenfeld, 19981], and fractals [Pentland, 1984]

In this dissertation, we are focus on a particular approach to texture analysis which is referred as the *rule-based* approach. The method generates symbolic descriptions of texture by first, deriving structural features from texture, and then generating covers (description) to fit texture data. Thus, it can be viewed as the form of model-based approach. The novelty of this approach lies in automatic generation of texture models (called the Multilevel Logical Templates) by the inductive learning process. Learned models are optimized to increase their descriptiveness for the texture concepts they represent.

1.2 Motivation

Intelligent behavior is characterized by the ability to learn and to self-organize. Such abilities include self-creation of knowledge through experience, self-improvement of control through practice, discovery of new concepts and relationships through observation and analogy, and reorganization of knowledge to perform a given task better. When we consider human or animal vision, it is clear that they acquire the ability to recognize and handle visual objects to a large extent through learning, and only partially

by genetic pre-programming. Yet, current computer vision techniques are almost devoid of learning capabilities. Further progress in computer vision will depend on the implementation of machine learning capabilities in vision systems.

Many vision systems of the past have been successful at processing images or extracting useful information from images. With the interests in image understanding, processing images goes beyond that to a point where the content of an image can actually be conceptualized, i.e., the system can describe what it sees in the image or scene. Until recently, most approaches to vision have been largely statistical in nature. They use mathematical formulas which transform the image into discernible objects. The primary processing stage in these approaches is the derivation of visual information. The research presented in this thesis is motivated by the belief that further progress in computer vision will depend not only on the derivation of visual information but on the intelligent manipulation of visual information as well. Symbolic learning techniques can enhance image understanding capabilities of computational vision systems by generating more expressive, conceptualized and meaningful descriptions of visual objects. Such descriptions can be intelligently manipulated to enhance their performance and utilization within the vision system.

Object recognition is one of the most studied areas of machine vision. The development of computer systems capable of recognizing objects within images is a long-term research problem of great practical significance. Several decades of research in this direction have produced many important results, but the progress has been slow. There is still no computer system that can reliably assign a generic class to natural objects on the basis of its image. One of the significant weaknesses of current computer vision systems is that they rely primarily on the structures programmed into them, and have very limited or no learning capabilities. Thus, they lack flexibility, adaptability, and cannot improve with experience. Intelligent behaviors, however, are characterized by the ability to adapt and self-organize according to the specifics of the environment and a given task. Humans adapt mainly through learning. Object recognition by humans deeply integrates learning with vision. This integration not only supports the basic capabilities of model acquisition, but it also supports the adaptability and robustness of object recognition in real-world noisy environments.

Although there have been many efforts to implement learning capabilities for object recognition, results are quite modest. In many cases, computer vision researchers still apply old pattern recognition techniques to the object recognition task. These techniques, developed in the 50s and 60s, do not represent the current state of the art in machine learning methodology. The most popular method, the minimization of Bayes risk [Duda and Hart, 1973], is a strong parametric method that assumes the distribution of features to be known *a-priori* . This condition cannot be fulfilled in most situations and this method does not perform well in the case of a complex attribute distribution [Pachowicz and Bala, 1991]. If feature distribution is irregular, then non-parametric methods are more suitable. These methods, however, have many disadvantages. For example, Nearest-Neighbors classifiers require large storage capacity and extensive calculations during the recognition phase. Pattern recognition methods are more applicable to static problems. Their extension towards incremental acquisition or evolution of class descriptions over time is extremely difficult. Moreover, these methods are not suitable to the integration of numeric and symbolic features in the acquisition of class descriptions.

Machine learning and cognitive science have been investigating computational models of human learning, and building machines capable of learning as humans do. The goal of early research in machine learning was to demonstrate that machines can learn simple concepts and classification procedures. In the past, researchers often ignored the characteristics of real world environments. Since then, significant progress in machine learning has been made. New powerful learning methods and tools have been developed, and their usefulness has been demonstrated in solving quite complex problems [Bergadano, Matwin et al., 1992]. The complexity of many problems, however, is very high and limits the broad application of currently available tools. This problem is particularly severe in computer vision. The creation of vision systems with learning capabilities requires the development of new learning methods, vision paradigms, and associated integration schemas. Learning technology provides a great challenge for machine vision, particularly in learning and representing object descriptions, advancing flexibility, and adaptability of machine vision [MLV-92, 1992].

New constructive induction learning tools can be useful in learning from vision data [Bala, Michalski et al., 1992]. The general concept of constructive induction includes any method that self-modifies the concept representation space during the induction process. Generating additional, problem oriented attributes is an important form of such self-modification of the representation space [Michalski, 1978]. In vision systems the key characteristic patterns within a hybrid attributional space can be detected by a constructive induction learning tool and defined as new attributes.

1.3 Thesis Overview

Chapter 2 introduces characteristics of sensory data and problems related to learning from such data. Presented characteristics include: noise in the data, complex problem spaces, large training sets, and a large number of classes.

The general goal of the research reported in this thesis is to investigate the inductive learning approach to texture recognition. The approach is based on the idea proposed by Michalski [Michalski, 1973]. The basic ideas behind the approach are presented in Chapter 3.

The approach has been implemented in the TEXTRAL learning system, and experimentally tested on texture recognition domains. Chapter 4 describes all phases of the learning methods in detail. Initial concept description optimization methods are also described in Chapter 4.

Experimental results of the TEXTRAL method are reported in Chapter 5. They include experiments with concept description optimization in learning from the first image level and experiments in the multilevel application of inductive learning to texture recognition.

The AQ-NT and AQ-GA optimization methods are presented in Chapter 6. This chapter also reports on experiments with these methods.

Related work is presented in Chapter 7. The TEXTRAL method represents an extension and improvement of our earlier method implemented in the TEXPERT system [Channic, 1988]. This system is briefly described in this chapter. A comparison of rule-based (TEXTRAL type) approach versus a k-NN method [Pachowicz, 1991] and the constructive induction method (the PRAX method) [Bala, Michalski, et.al., 1993] are also presented in this chapter.

Finally, Chapter 8 summarizes accomplished work, and discusses limitations and possible future research.

Chapter 2

2. Learning from Sensory Data

This chapter introduces characteristics of sensory data and problems related to learning from such data. Characteristics that can be expected in the vision data include: noise, complex representation spaces, large training sets, and large numbers of classes.

2.1 Noise in the Data

Sensor-driven characteristics of visual objects are rarely noise free and most often quite noisy. "*The visual world is noisy. Even well posed visual computations are often numerically unstable, if noise is present in both the scene and the image. Scenes are usually corrupted by "noise" coming from various sources (dust, fog, sun glitter, etc.). The image formation process introduces additional noise. As a result, many problems which theoretically have unique solutions become very unstable in the presence of input noise.*" [Rosenfeld, Aloimonos et al., 1990].

To explore the application of learning techniques to vision domains, it is, therefore, very important to develop approaches that are successful despite a high level of noise in the data [Chien, 1991]. In traditional classification problems, noise comprises non-systematic errors in the values of attributes or class information [Quinlan, 1986]. There are at least three types of noise that must be dealt with: (i) *uncertainty*, (ii) *error*, and (iii) *imprecision*.

Uncertainty occurs when the correct value of an attribute cannot be determined. *Error* occurs due to a misreading or incorrect coding of the information. For continuous (or fine grained) attributes another form of noise is *imprecision*; the best obtainable measurement for the value is imprecise. Although noise due to *uncertainty* and *error* is problematic, there is no method of avoiding *imprecision*. Therefore any learning algorithm must be able to handle some noise in those attributes. Additionally, learning tools have to deal with another source of noise; *misclassification error*. Traditionally, we assume that the training data is perfectly classified. However, in the case of vision domains (e.g., image interpretation) data pre-classification is also very difficult and imprecise. The elimination of such misclassification error is therefore essential for learning tools applied to computer vision problems.

It is also useful to differentiate between two levels at which noise can occur: the *input level* and the *algorithmic level*. On the input level, we are concerned with the "raw" data which is input to the system, and the focus is on acquisition errors. Acquisition errors are variations which are caused by the process of gathering the input data. Some of the variations which are caused by the properties of the camera can also be modeled. Other variations, on the other hand, cannot be modeled so easily. Consider, for example, a camera in which some of the sensing elements are defective in such a way that they either record a constant value (e.g., black or white) or an arbitrary value with an unknown probability distribution. Even if the fact that this problem exists is taken into consideration, information is still lacking on (i) the probability of its occurrence, (ii) the fraction of the defective pixels, (iii) the (unknown) probability distribution of the errors, and probably most importantly (iv) the spatial distribution of the defective pixels over the frame (e.g., are they randomly distributed or clustered). Considering the misclassification error, modeling of automatic or human interpretation of data is almost impossible.

Another source of input level noise may be due to environmental conditions. For example, noise in vision data can be due to the masking of object properties by other objects (clutter), from object irregularities, by the variability of object characteristics, or by an incorrect classification given by a teacher. All of these noise sources are generally very difficult to model in that the distribution of the variations is usually unknown.

Noise on the algorithmic level occurs at a higher level than the types discussed above. In this situation an algorithm is viewed as being composed of modules each of which can be considered individually as an algorithm. The inputs to higher level modules are the results produced by lower level modules. Thus, noise which was not handled correctly by lower modules can propagate errors to the higher levels. As before, variations caused by noise on the algorithmic level are difficult to model. If algorithmic noise could be modeled then it could be corrected and not propagated.

Noise on the input and algorithmic level is caused by processes that are difficult to model a priori. Although there might exist models that govern these processes, these models are unknown a priori to the system. In engineering literature, theoretical modeling of noise effects on the system behavior is typically done using "white noise" (i.e., white Gaussian noise). In practical situations, however, we have to deal with a combination of different classes of noise (including salt-and-pepper noise, pink noise) of very complex distribution.

As indicated by Quinlan [Quinlan, 1986], regardless of the source, noise can be expected to affect the formation and use of classification rules. Inductive learning systems must perform some form of generalization in order to anticipate unseen examples. Ideally, a concept description generated by an inductive learning system should cover all examples (including unseen examples) of the concept (positive examples) and no examples of all other concepts (negative examples). Thus, most inductive learning systems generate a complete and consistent concept description which covers all positive examples and no negative examples [Pratt, Faugeras et al., 1978] . In the case of noisy data, complete and consistent descriptions are problematic because multiple concept descriptions can partially overlap in the attribute space. This is so, because attribute noise skews the distribution of attribute value from the correct value. Because of the existence of noise in the vision data, some positive examples are noise, that is, they are actually negative examples. We call such examples "positive noisy examples". These positive noisy examples are covered by the complete and consistent description. These examples, however, should not be covered. On the other hand, some negative examples can be noise, i.e., they are actually positive examples. Such negative examples are referred to as "negative noisy examples". Negative noisy examples are incorrectly left uncovered.

2.2 Complex Problem Spaces

Another expected characteristic of sensory data is a highly complex attribute distribution. Figure 2-1 shows selected attribute distributions obtained in experiments with noisy texture data [Pachowicz and Bala, 1991]. An approximated normal distribution is also shown. The upper diagram (Figure 2-1 a) shows a sample of a multi-cluster distribution for one class. The lower diagram (Figure 2-1 b) shows an extremely non-normal attribute distribution of an attribute that lacks the discriminating power (no sharp peak and no major clusters through the wide spectrum of attribute values). The distribution was obtained by extracting the attribute values from one texture class. The distribution of the same attribute, especially for discriminating it from other classes, may be quite appropriate for another class. Although the presented distributions are related to single attributes, the attribute space is multi-dimensional and its complexity increases significantly if distributions of single space components are complex.

Figure 2-2 depicts examples of two classes distributed over the two attributes cross-section of the representation space obtained from vision data. A complex, overlapping distribution of examples of these classes can be observed. Learning from such complex overlapping data generates complex class descriptions. In the case of a DNF description (disjunctive normal form), a large number of disjuncts (rules) is needed to cover examples of a given class. Similarly for decision tree learning, complex and large decision trees will be generated.

Recent research shows [Michalski, 1986] [Weiss and Indurkhya, 1991] that while an increasingly complex description can usually be generated to better cover training examples, the predictive accuracy of this description for new unknown examples may be inferior to a simpler, less complex description. For example, a fully expanded decision tree or highly disjunctive rule set may cover training samples completely, but a smaller tree or rule set, with a larger apparent error on the training examples, may be more accurate in its prediction for new unknown examples. Complex problem spaces, characteristic of sensory domains, inhibit the generation of simple, refined descriptions.

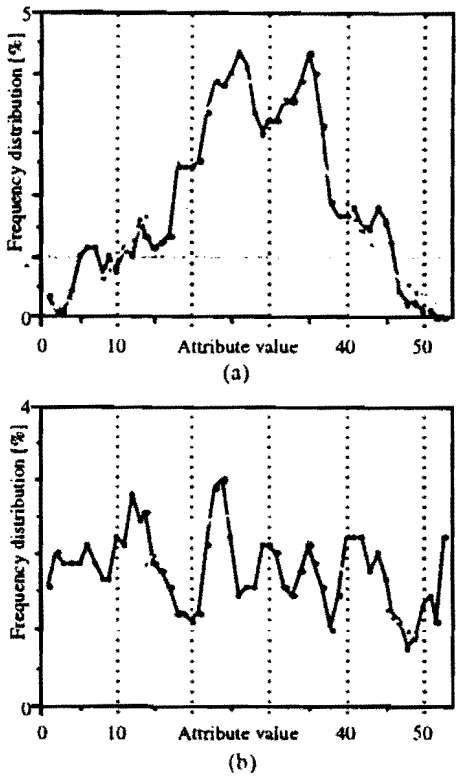


Figure 2-1: Examples of complex distributions of texture attributes extracted from a single class

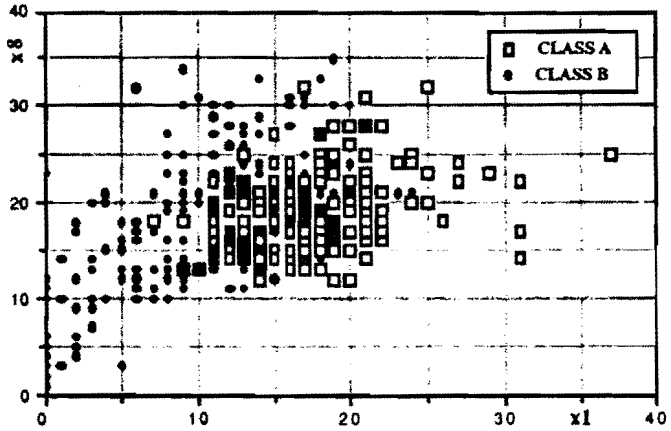


Figure 2-2: A cross-section over two vision attributes *

* x1 and x8 are Law's masks attributes, [Laws, 1980] , (Section 4.3).

2.3 Large Training Sets and Multiclass Environments

As different learning algorithms have been developed and refined, the size of the training sets attacked has grown. In some situations (e.g., low level vision) learning from vision data can involve examples with hundreds of attributes and thousands of examples and hundreds of classes. New learning tools are needed to handle such a large sets of training examples. Recently, some researchers reported experiments involving large training sets. For example, [Dietterich, Hild et al., 1990] report on a decision tree technique applied to training set of about 20,000 instances concerning the pronunciation of English. The Space Shuttle datsets described in [Catlett, 1991] comprise hundreds of thousands of examples: NASA's archives of flight data contain more than 150 million training examples. Faced with such an enormous training set, one might ask the reasonable question whether the whole set should be used or some part of it. If a small sample turns out to give results as good as the full set, learning from a full set is an unnecessary expense. Very large sets can also yield descriptions that are too large (highly disjunctive rule sets, large decision trees) to be effectively utilized. The AQ-NT method described in this thesis reduces the complexity of concept description by reducing the size of training set.

Very often a user requires a comprehension of the obtained description. The size of generated decision trees and rule sets is important for comprehensibility of the description. If the size of the description becomes too large, human experts will be unable to understand it. Figure 2-3 depicts the description complexity (as the average number of rules per class) for different number of classes and examples per class obtained from texture data using the AQ learning method.

There are different important factors that should be taken into consideration in multiclass environments. First, how do multiclass learning methods compare in terms of their ability to classify unseen examples correctly? Second, are some methods more difficult to train than others (i.e., do they require more training examples to achieve the same level of performance?) Third, should characteristic or discriminatory descriptions be generated?

Learning time is another important factor that should be taken into consideration in multiclass environments. The growth of learning time is largely due to the need to process large number of examples of different classes. For examples with continuous attributes some method is required to quantize the continuous values, turning them into a small set of ordered discrete values, which allows faster processing of the data. Some methods use sampling and statistical inference techniques to eliminate the size of the data. Other approaches are also suggested, such as constructive induction to improve the attributes used to express the concepts.

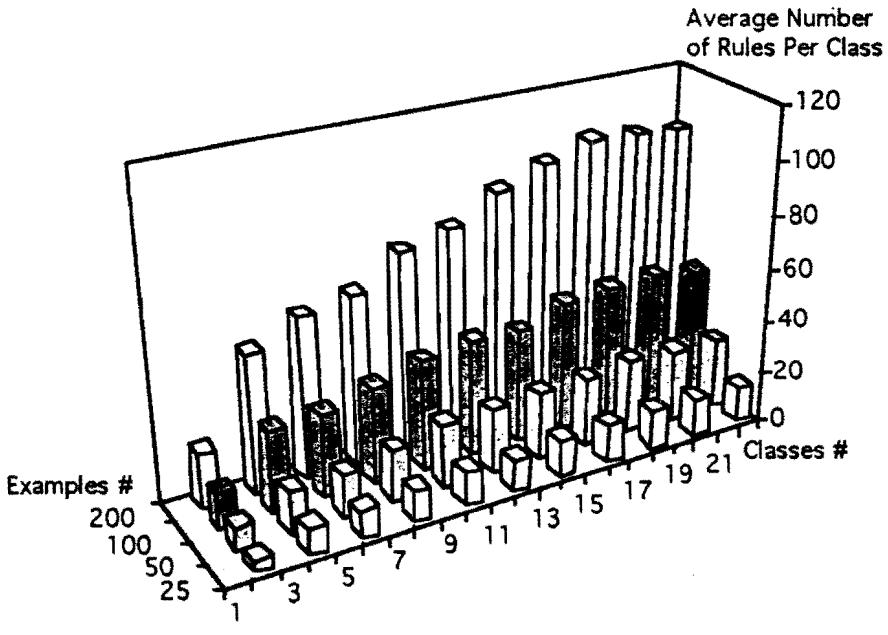


Figure 2-3: Average number of rules per class

Chapter 3

3. General Approach

The developed approach, called "Multilevel Logical Templates" (MLT) (the approach was originally proposed by Michalski [Michalski, 1973] and was initially applied using the ILLIAC III image recognition computer facilities) aims at automatically determining texture class descriptions from texture samples. This approach was implemented as the TEXTRAL system. The next chapter describes the TEXTRAL system in greater detail. The basic step in the MTL approach is an iterative (multilevel) application of symbolic inductive learning to generate texture rules. These rules serve as "logical templates" that are matched against examples extracted from unknown samples of texture classes.

The basic idea behind the MLT approach can be explained as follows (Figure 3-1). Given an image with labeled samples of different textures (these samples are attribute vectors), the learning system generates sets of rules describing samples of different textures (Logical Templates). In the iterative mode of the method (not depicted in Figure 3-1) these rules are used to transform this image to a "symbolic" image, in which picture elements are labels of corresponding texture areas. New sets of rules (next level template) can be learned from the symbolic image. A set of rules with extraction operators (i.e., operators that are used to extract vector of attribute values from different position in the textural area to characterize a texture class) is called "texture signature".

To recognize an unknown texture sample, the system matches it with all candidate texture descriptions. This is done by applying decision rules to the events in the sample. For each event, the class membership (texture class) is determined, the recognition

accuracy for the sample is computed. The assignment of the sample to a given decision class (texture) is based on determining which of the candidate classes gets the majority (or) plurality of votes. Thus, even if some events in the sample are incorrectly recognized, the classification of the sample may be correct. Experiments described in this thesis report on recognition accuracy of the first level set of rules (the first level templates). Some experiments with multilevel sets of rules (multilevel template) are also presented.

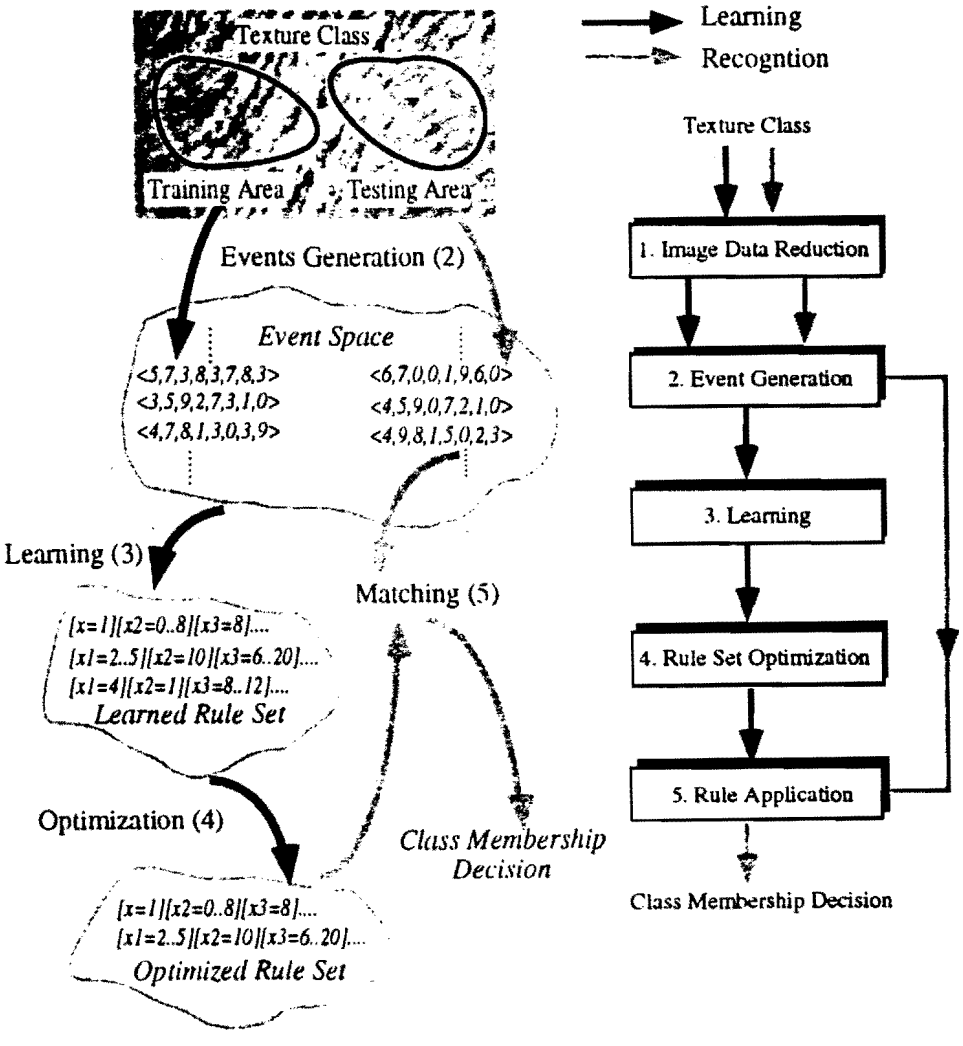


Figure 3-1: An illustration of the general approach

There are following phases of the TEXTRAL learning method (Figure 3-1):

(i) image preprocessing (volume optimization),

(ii) training events generation (selection of texture samples, determining attributes, and formulating training examples),

(iii) inductive learning of texture rules (generation of "logical templates"),

(iv) texture rules optimization, and

(v) recognition of unknown texture samples.

The first phase of the process reduces the spatial resolution of the image. The second phase extracts a set of spatial texture samples, called *events*, from classified texture regions (Module 2 in Figure 3-1). An event is a vector of attribute values that represent different image (texture) features. Attributes values are obtained by using two-dimensional convolutions and simple moving-average techniques. Additional attributes can be determined through the process of constructive induction [Wnek and Michalski, 1991] [Bala, Michalski et al., 1992].

There are many possible attributes that could be determined to characterize textures. The most desirable are those that define a description space in which points corresponding to the same texture class constitute easily describable clusters. In the texture recognition domain these attributes may fall into one of three categories: neighboring gray-level values, statistical measurements, and convolution filter outputs. Sets of events extracted from texture classes to be learned are used as training examples (events).

3.1 Learning Method

Texture rules are determined using the AQ-15 method for inductive concept learning from examples (Module 3 in Figure 3-1) [Michalski, 1986]. The rules learned by the AQ method are represented in VL_1 (Variable-Valued Logic System 1); [Michalski, 1972]. Advantages of this representation are that it is amenable for parallel execution, and

easy to interpret conceptually. A concept description is a logical expression in disjunctive normal form associated with a decision class (here, a texture class). Each conjunction in this expression together with the associated decision class can be viewed as a single decision rule. The conjunctions (serving as condition parts of the rules) are logical products of elementary conditions in the form:

[L # R]

where:

L, called the *referee*, denotes an attribute.
R, called the *referent*, is a subset of values from the domain of the attribute L.
is one of the following relational symbols:
=, <, >, >=, <=, <>.

Each rule is assigned two parameters: "t" (for "total weight")—measuring the total number of positive training examples covered by the rule, and "u" (for "unique weight")—measuring the number of positive examples covered by the given rule and not covered by any other rule for the given decision class. Here is an example of an AQ-15 decision rule:

[Class=1] ::> [x2=1][x4>3][x6=1..7]: (t=6, u=2)

This rule covers 6 examples of Class 1, out of which 2 are covered only by this rule, and not by any other rule for this class. In the case of texture rules, x_i are attributes characterizing a texture sample (in our experiments we used primarily 8x8 windows). The above rule is satisfied, if attribute x2 takes value 1, attribute x4 has value greater than 3, and attribute x6 takes value between 1 and 7.

As mentioned earlier, a description of a texture class can be viewed a set of such rules (a "ruleset"). In such a ruleset, individual rules are ordered according to the decreasing values of the t-weight. As mentioned earlier, the extraction operators with the logical description generated by the AQ algorithms is called a texture signature. Figure 3-2 is an example of the first level texture signature. This texture description is was optimized by truncating its less significant rules. The x1 to x8 operators are: x1 -

Laplacian edge operator, x2 - the Frequency spot, x3 -the horizontal edge operator, **x4** - the vertical edge operator, - is the horizontal V-shape operator, x6 - the vertical V-shape operator, x7 - the vertical line operator, and x8 - the horizontal line operator.

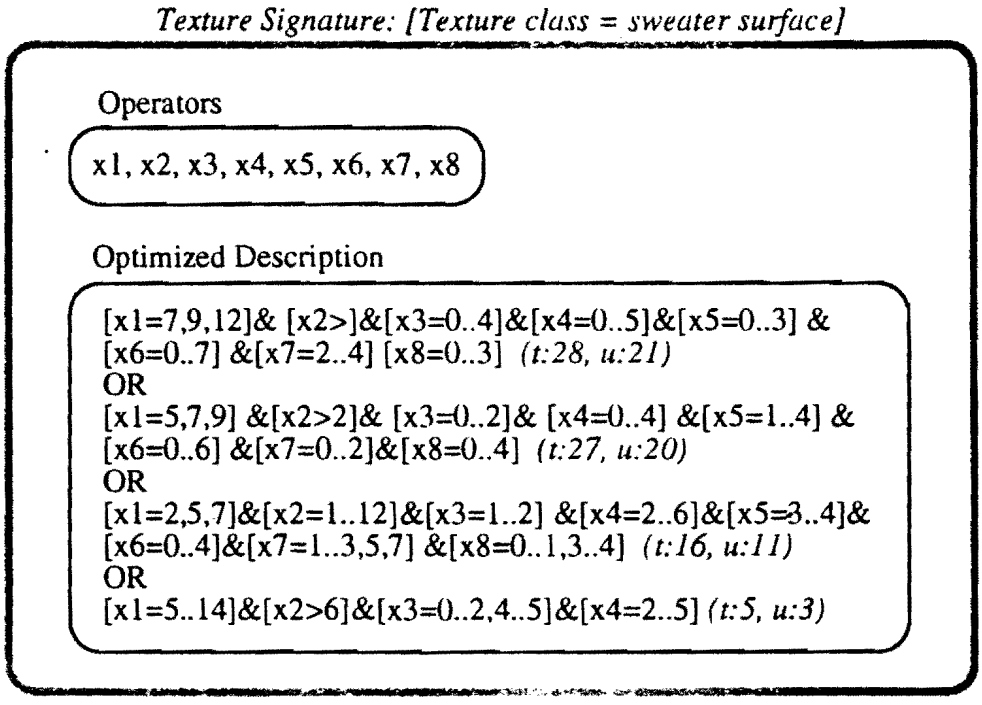


Figure 3-2: An example of texture signature

The method uses "truncated" descriptions of texture classes. A truncated description is obtained by the removing from the initially generated rules the ones with a very low t-weight. The reason for this is that rules with a low t-weight can be viewed as insignificant, or as representing noise. It has been discovered experimentally that truncated descriptions often give a higher texture recognition performance than non-truncated descriptions. Since truncated descriptions are also simpler, then such a truncation process is highly desirable. A detailed study of this phenomenon (in the

context of non-vision applications) have been described in [Bergadano, Matwin et al., 1992].

3.2 Description Optimization

One of the most important problems in application of machine learning to vision data is the influence of noise. Image data is corrupted with noise, and consequently, learned concept descriptions contain noisy components. The main objective of optimization methods (Module 4 in Figure 3-1) is to decrease the influence of noise on learned descriptions according to some performance measure.

In the rules optimization phase of the TEXTRAL method concept descriptions are analyzed and modified. Different methods for concept description optimization and their experimental validations are described in this thesis.

3.3 Recognition Method

The learned texture descriptions are generalizations of the observed texture events (i.e., attribute-value vectors characterizing window-size texture samples). Therefore, they can be used to classify unobserved texture samples (Module 5 in Figure 3-1). There are two methods for applying the descriptions for recognizing the class membership of an event: the *strict* match and the *flexible* match.

In the strict match, the system tests whether an event strictly satisfies (the condition part of) a rule. The satisfied rule determines the classification decision. In the flexible match, the system computes a degree of match between the event and candidate rules. The degree of match can vary in the range from 0. (no match) to 1.0 (complete match). The rule with the highest degree of match determines the classification decision.

To explain the calculation of the degree of match, assume that a recognition rule contains a condition $[x = a_k]$. If the domain of the attribute x is a set of n numerical

values $<a_1,a_2, ...,a_n>$, and an event includes the statement $[x=a_i]$, the normalized degree of match between the rule and the condition in the event is defined as:

$$1 - (| a_j - a_k | / n)$$

If the condition has several values in the referent (on its right-hand-side), the value closest to a_k is used. The degree of match between a rule containing several conditions and an event was computed as the average of the degrees of match between the conditions and the event conditions.

The degree of match between a class description (which may have several rules) and a given testing event (an example) is determined as the maximum of the degrees of match between individual rules in the description and the event. The description with the highest match among classes determines the recognition decision. The measure of recognition accuracy of a rule when applied to a set of testing events is the percentage of the number of correctly classified test events to the total number of testing events in the set.

The recognition phase of the MTL method is described in greater detail in Section 4.5 of the next chapter.

Chapter 4

4. The TEXTRAL Learning Method

This chapter describes the TEXTRAL method in detail. Different techniques that are part of this method are reported in: [Michalski, 1973], [Bala and Michalski, 1991] (the MTL approach), and [Bala and Pachowicz, 1991].(the description optimization).

The process of learning such texture descriptions consists of the following phases:

(i) image preprocessing (volume optimization),

(ii) training events generation (selection of texture samples, determining attributes, and formulating training examples),

(iii) inductive learning of texture rules (generation of "logical templates"),

(iv) texture rules optimization, and

(v) recognition of unknown texture samples.

4.1 Processing Texture Data

This section presents image processing stages applied in order to transform raw image data of texture into texture characteristics coded by a vector of attributes.

4.1.1 Image Data Used by TEXTRAL

Two sets of textures were used in experiments with the TEXTRAL system. One set, a twelve textures set, was acquired from the Brodatz Texture Depository [Brodatz,

1966] (Figure 4-1) and another one, a six textures set, from sweater surfaces (Figure 4-2).

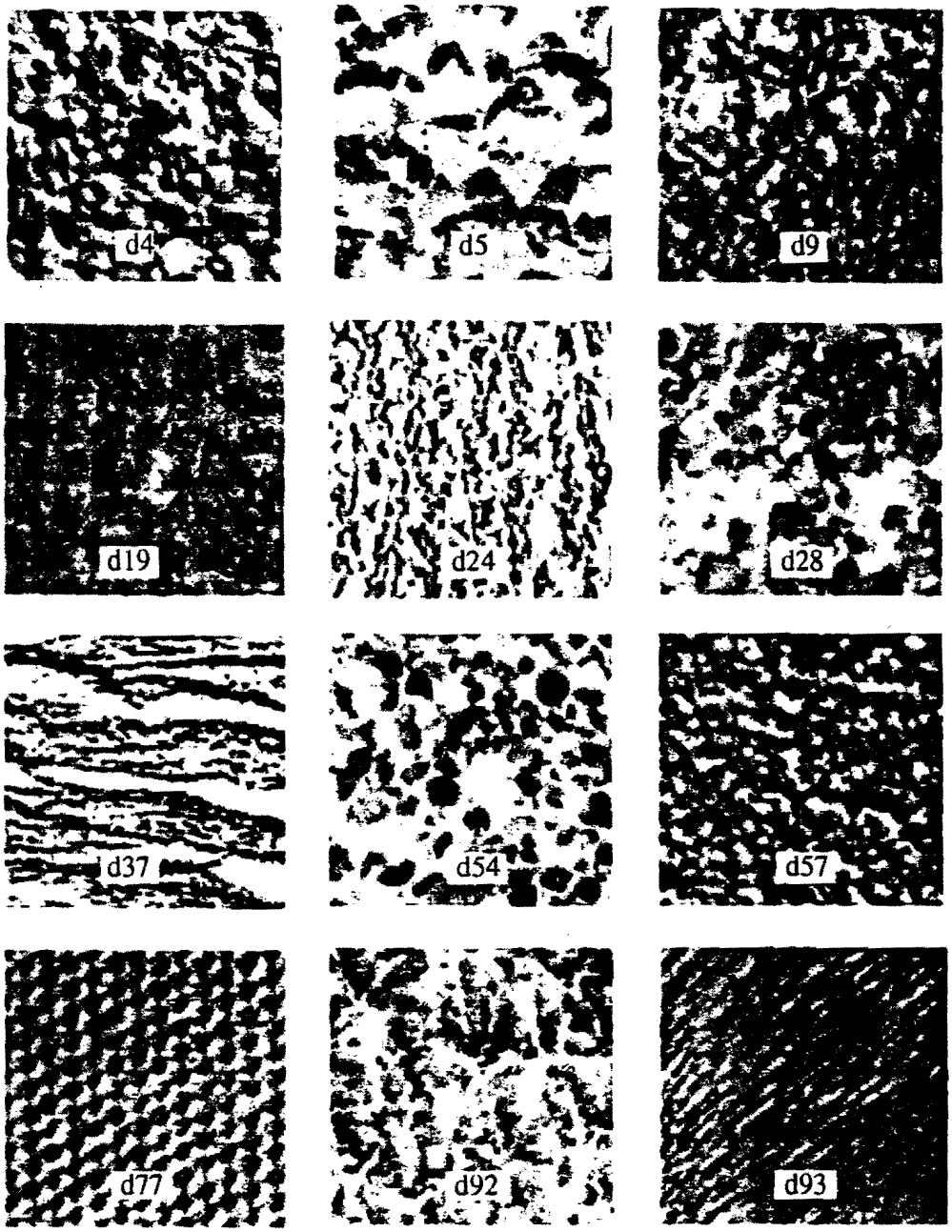


Figure 4-1: Textures from the Brodatz's set of textures

Figure 4-2: Sweater surface textures

Twelve textures in the first set are: pressed cork (d4), expanded mica (d5), grass lawn (d9), woolen cloth (d19), pressed calf leather (d24), beach sand (d28), water (d37), beach pebbles (d54), handmade paper (d57), cotton canvas (d77), pigskin (d92), and fur (d93). For most textures, the illumination was regular and the resolution was fixed.

However, several textures presented in Figure 4-1 (i.e., grass lawn, water and beach pebbles) were taken through an angled projection to the object surface. Thus, these textures were affected by the irregularity of resolution.

The selection of twelve textures classes was motivated to include both visually different textures (e.g., pressed calf leather (d24), water (d37), fur (d93)), and also those textures that are very similar and do not have directionality feature (e.g., pressed cork (d4), beach sand (d28), beach pebbles (d54)). Another motivation for this selection is the interests of this research to test learning approach on the attribute space that includes highly separated and/or also overlapping clusters of texture characteristics.

Each texture was coded as an image of 512 by 512 pixels, where a single pixel was quantified to 256 gray levels (samples of twelve textures presented in Figure 4-1 are approximately 60 by 60 pixels). Each image was divided into two large sections. The first half of an image was used to extract learning examples while the second half was used to extract testing examples.

4.1.2 Reduction of Spatial Resolution

In the first phase of the TEXTRAL method an image data is prepared for attribute extraction and training set formation. The spatial resolution of an image is reduced. Images acquired for experiments by camera are digitized in 512 lines and 512 pixels per line. The spatial resolution is reduced by partitioning the digitized image into nonoverlapping neighborhoods of equal size and shape and replacing each of those neighborhoods by the average pixel densities in that neighborhood. The resolution is reduced with respect to the size of the extraction operators (section 4.3 describes extraction operators). The extraction operator size is related to the intersample spacing in the extraction process. In experiments presented in this paper, 5 by 5 extraction operators were used. To determine the lowest resolution for a given intersample spacing a second order statistic, in form of the gray level co-occurrence matrix [Haralick, Shanmugan et al., 1973] was used. The gray level co-occurrence matrix method is based on the estimation of the second-order joint conditional probability density functions $f(i,j|d, \phi)$. Each $f(..)$ is the probability of going from gray level i to gray level j, given the

intersample spacing d with direction ϕ. The estimated values can be written in matrix form, the so-called co-occurrence matrix.

From a co-occurrence matrix a number of features can be derived. If a texture is coarse, and d is small compared to the sizes of the texture elements, the pairs of points at separation d usually should have similar gray levels. This means that the high values in the matrix should be concentrated on or near the main diagonal. For a fine texture, if d is comparable to the texture element size, then the gray levels of points separated by d should often be quite different, so that the high values in the matrix should be spread out relatively uniformly. In different experiments [Bala, 1990] the best match between the intersample spacing $d=5$ and different spatial resolution levels was obtained after one consolidation operation. Thus, the spatial resolution of textures depicted in Figure 4-1 and 4-2 was reduced from the 521 by 512 to 256 by 256 pixels. Because textures are quite fine, further reduction of the spatial resolution is blurring important characteristics of textures.

4.3 Texture Attribute Values Extraction

Many possibilities exist for computing features to recognize texture. The best features will be those which define an attribute space which most easily lends itself to the partitioning performed by the learning algorithm. Most of features fall into one of three categories: neighboring gray-level values, simple statistics, and convolution filter output.

Neighboring Gray-Level Values are the simplest attributes. They consist only of the gray-level values of pixels neighboring the pixel with which the event is associated. For example, one could take the value for each immediate neighbor of a pixel to construct an event with eight attributes for each pixel. The first attribute would correspond to the value of the pixel to the central pixel's upper left, the second attribute would correspond to the pixel directly above the central pixel, and so on.

Simple statistics are linear combinations of the neighboring pixel values. Some examples of such statistics are sum of values, mean pixel value, maximum pixel value, and minimum pixel value. One of the most often used statistics is a second order

statistics, in form of the gray level co-occurrence matrix [Haralick, Shanmugan et al., 1973]. The gray level co-occurrence matrix method is based on the estimation of the second-order joint conditional probability density functions. The main advantage of the use of these statistics is that they incorporate regional information about the pixel in one feature with only minimal additional computation.

The third type of feature of benefit to a texture learning system are those that are calculated using filters or convolution operators. A convolution operator can be thought of as a coefficient template — a grid which is centered over a pixel in the image where each coefficient in the template multiplies the corresponding pixel value in the grid and the sum of these products is the value of the operator at the central pixel. The convolution method is more accurate than gray level co-occurrence methods. It is local, operating on small image windows in much the same manner as human visual system. It can be invariant to changes in luminance and contrast. Thus, a convolution method was selected in this research for attribute extraction.

4.3.1 Analysis of Local Subset of Pixels

The most frequently used methods of texture feature extraction are based on the analysis of a local subset of pixels. This subset is defined by a small window in order to derive local characteristics of covered pixels. There are many methods applied to derive local characteristics of pixels. Since the development of low-level image processing techniques is not the scope of this research, well known and well performing Laws' energy filters [Laws, 1980] , were applied to extract texture characteristics. The Laws' energy filter are convolution operators. The extraction method using these convolution operators consists of the following two steps:

(1) the extraction of local micro-characteristics of raw texture data incorporating specially designed energy masks (filters) detecting local pixel variations over a small window.
(2) the computation of local macro-statistics applied to derive statistical measures of filtered images over a larger window.

In the first step, the original image is filtered with a set of small convolution masks, typically 5 by 5 masks with integer coefficients. The filtered images are then processed with a nonlinear "local texture energy" filter. This is simply a moving-window average of the absolute image values. Such moving-window operations are very fast. The best window size depends on the size of image texture regions.

Methods based on a similar approach were widely applied [Unser and Eden, 1989], [Hsiao and Sawchuk, 1989] to texture feature extraction, and they provided quite good discriminating power when compared with other methods [Du Buf, Kardan et al., 1990]. Moreover, this class of methods is easily implemented on parallel architectures and these methods can be run in real-time.

A slightly modified Laws' method of texture feature extraction was used in experiments [Pachowicz and Bala, 1991]. The modification included (i) the extension of a feature list from four to eight features extracted from raw image data, and (ii) the reshaping of a square window used to compute local macro-statistics from filtered texture images.

The proposed set of energy masks $M=\{M_i\}$ (depicted in Figure 4-3) consists of four 5x5 masks (i.e., R5R5, E5L5, E5S5, and L5S5*). This set includes one rotation invariant mask (i.e., R5R5) and three masks that are sensitive on texture directionality (i.e., E5L5, E5S5 and L5S5).

A set of mask was extended by adding three directionality sensitive masks rotated by 90 degrees (i.e., masks: L5E5, S5E5 and S5L5). A 3x3 Laplacian filter (i.e., mask S3S3) was also added to perceive texture roughness of lower resolution. A geometrical interpretation of masks is depicted in Figure 4-4. Masks are shown as the functions of two variables, x and y, representing a position in a 5 by 5 grid. The function values represent mask's coefficients.

All masks of a size $(2a+1)x(2a+1)^{\dagger}$ pixels are applied to transform an input image f(j,k) into a set of images of local micro-characteristics $G=\{g_i\}$ through the convolution operation; i.e.,

---

* Naming after [Laws, 1980]

† a=2 for masks depicted in Figure 4-3

$$g_i(j,k) = \sum_{m=-a}^{a} \sum_{n=-a}^{a} M_i(m,n) * f(j+m,k+n)$$

for $i = 1,\ldots,8$

1	-4	6	-4	1
-4	16	-24	16	-4
6	-24	36	-24	6
-4	16	-24	16	-4
1	-4	6	-4	1

R5R5

-1	-4	-6	-4	-1
-2	-8	-12	-8	-2
0	0	0	0	0
2	8	12	8	2
1	4	6	4	1

E5L5

-1	0	2	0	-1
-2	0	4	0	-2
0	0	0	0	0
2	0	-4	0	2
1	0	-2	0	1

E5S5

-1	0	2	0	-1
-4	0	8	0	-4
-6	0	12	0	-6
-4	0	8	0	-4
-1	0	2	0	-1

L5S5

1	-2	1
-2	4	-2
1	-2	1

S3S3

-1	-2	0	2	1
-4	-8	0	8	4
-6	-12	0	12	6
-4	-8	0	8	4
-1	-2	0	2	1

L5E5

-1	-2	0	2	1
0	0	0	0	0
2	4	0	-4	-2
0	0	0	0	0
-1	-2	0	2	1

S5E5

-1	-4	-6	-4	-1
.0	0	0	0	0
2	8	12	8	2
0	0	0	0	0
-1	-4	-6	-4	-1

S5L5

R5R5 - Frequency Spot Operator (F-Spot)

E5L5 - Horizontal Edge (H. Edge)

L5E5 - Vertical Edge (V. edge)

E5S5 - Horizontal V-Shape (H. V-Shape)

S5E5 - Vertical V-Shape (V. V-Shape)

L5S5 - Vertical Line (V. Line)

S5L5 - Horizontal Line (h. Line)

S3S3 - Laplacian

Figure 4-3: A set of convolution masks

In the second step, Laws proposed [Laws, 1980] to compute the average absolute value over a larger window applied to each filtered image g_i separately; i.e.,

$$h_i(j,k) = 1/(\#S)(\sum \sum)_S |g_i(m,n)|$$

$$for \quad i=1,\ldots,8$$

where S corresponds to the local averaging window of the #S number of pixels. In this way, local averaging computes a statistical measure that can be considered a "texture energy measure". These statistics are considered a fast approximation to the standard deviation if a zero mean matched filter is applied. The computation of macro-statistics depends both on the shape and size of the moving window, and also on the weight coefficients within the window. Each pixel within the moving window has the same weight equal to 1, i.e., its influence on the computation of a given feature does not depend on the distance from the center of the moving window.

The large window improves the stability of the texture feature over the homogeneous area. Originally, Laws applied a square moving window of 15x15 pixels. Such a large window, however, causes classification errors when the method is applied to the texture segmentation problem. These errors occur frequently on the borders between different texture areas. Relatively small texture areas are blurred or even not distinguished at all for larger averaging windows. On the other hand, a small averaging window causes extracted texture features to be very noisy which has a significant negative influence on the acquisition, representation, and recognition of textures. Modifications to the size of the moving window have already been proposed ([Tomita and Tsuji, 1977], [Hsiao and Sawchuk, 1989]) to deal with the application of the texture energy method to the texture segmentation problem. This research followed such an approach choosing several window sizes (i.e., windows of radius R≈3.5, 5.5 and 7.5 in Figure 4-5) and reshaping the window from square to circular. A circular window stabilizes the distance variation from the central pixel to border pixels of the averaging window.

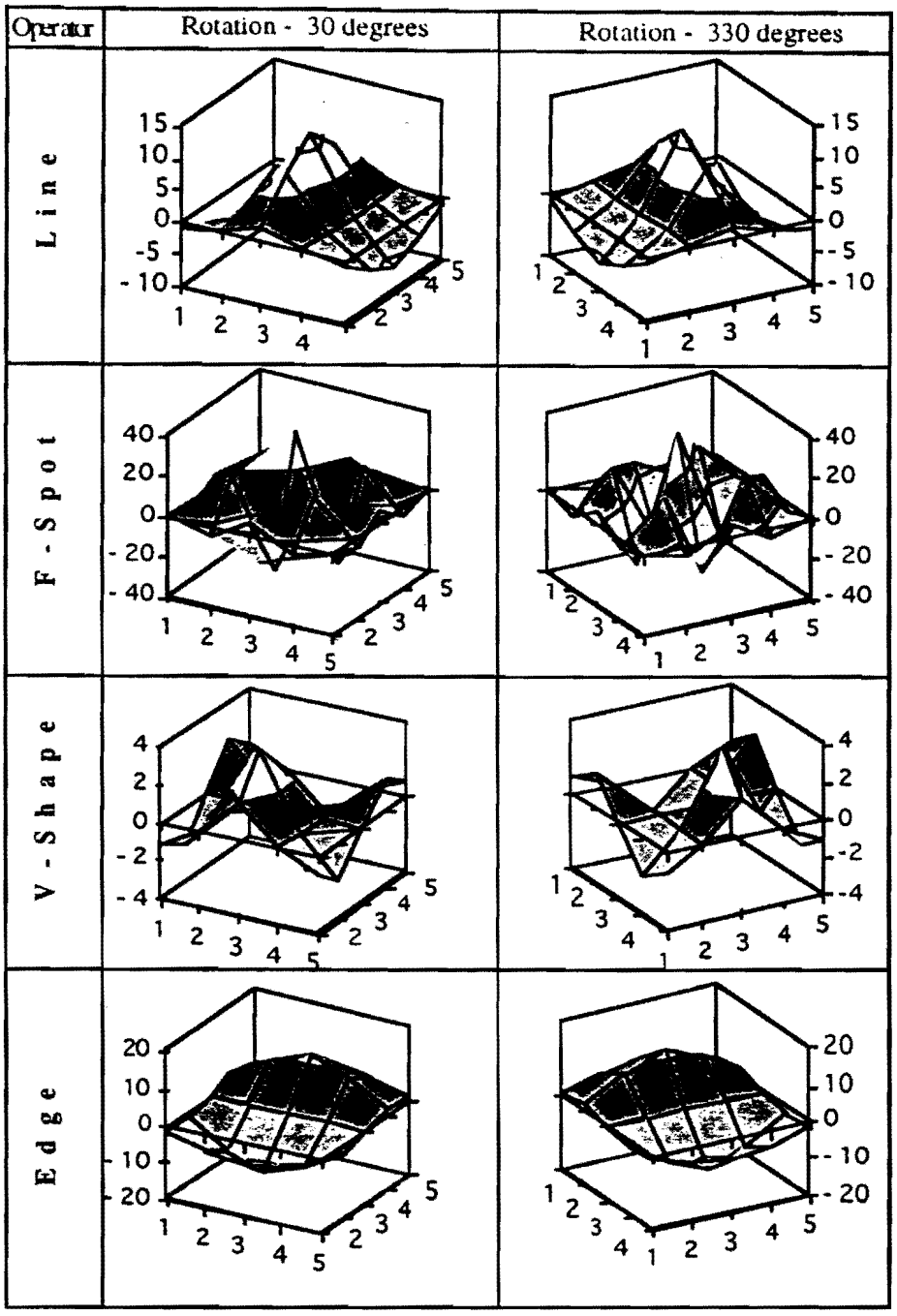


Figure 4-4: Geometrical interpretation of convolution masks

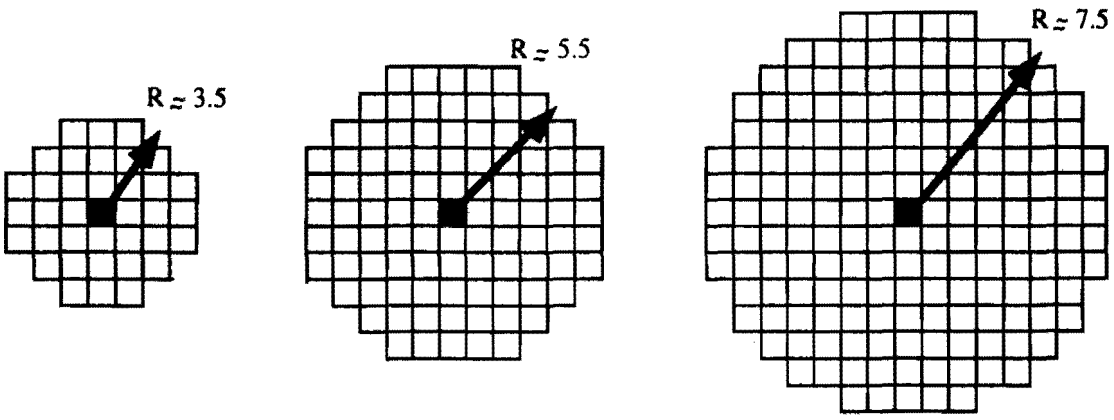


Figure 4-5: Circular averaging windows of different radius

4.3.2 An Example of Texture Attribute Extraction

Figure 4-6 and following Figure 4-7 and Figure 4-8 show an example of attribute extraction using Laws' masks. Areas on the left represent convolved and smoothed textures. A corresponding histogram for each area is also depicted. The histogram shows a percentage of pixels in the area with a given gray-level value.

Figure 4-9 shows few pixel positions with their corresponding intensities in two areas obtained by convolving and averaging the original image sample with the F-Spot and H. V-Shape operators. All eight attributes values are extracted in the same way as depicted in Figure 4-9.

4.3.3 Selecting And Coding Texture Events

Acquired texture features are grouped for each (j,k) pixel of the input image into a vector of learning attributes $\underline{h}(j,k)=(h_1, h_2, ..., h_8)$, where h_i is a integer number greater or equal to zero. Because the number of possible values of each attribute is high,

an attribute coding is applied to decrease this number to an acceptable range for the family of AQ programs (see next section). The AQ programs allow a single attribute a range of only 58 levels.

Attribute coding provides a good opportunity to compress data through the generalization of attributes values into symbolic intervals. Such generalization must be designed carefully in order to keep the discriminating power of acquired texture concepts. The decrease in discriminating power can occur when the number of intervals is low, there are textures both highly different and very similar, and the number of texture classes is large.

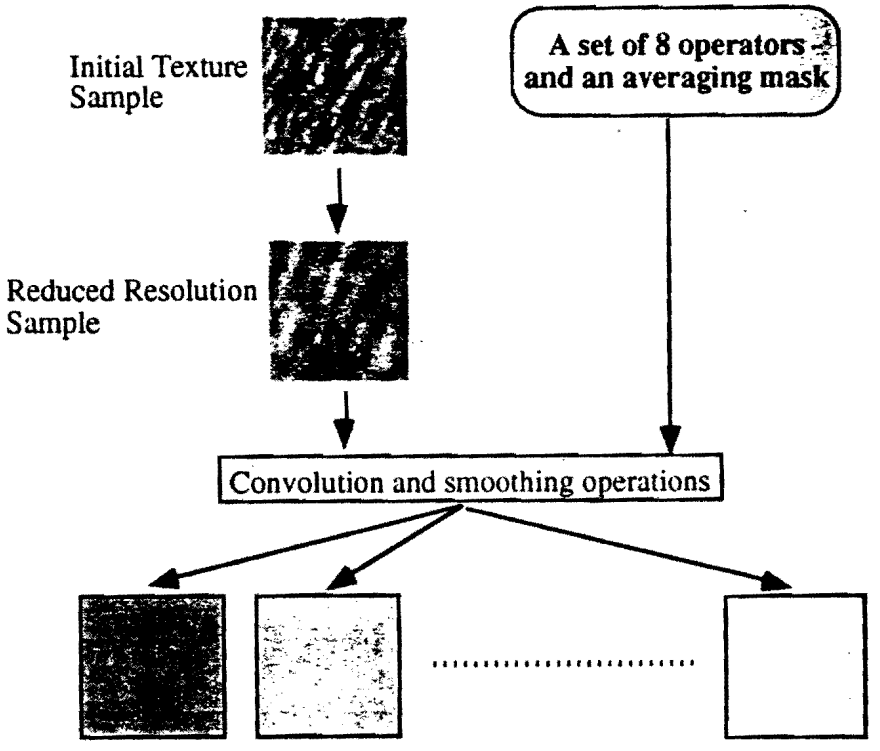


Figure 4-6: An example of attribute extraction

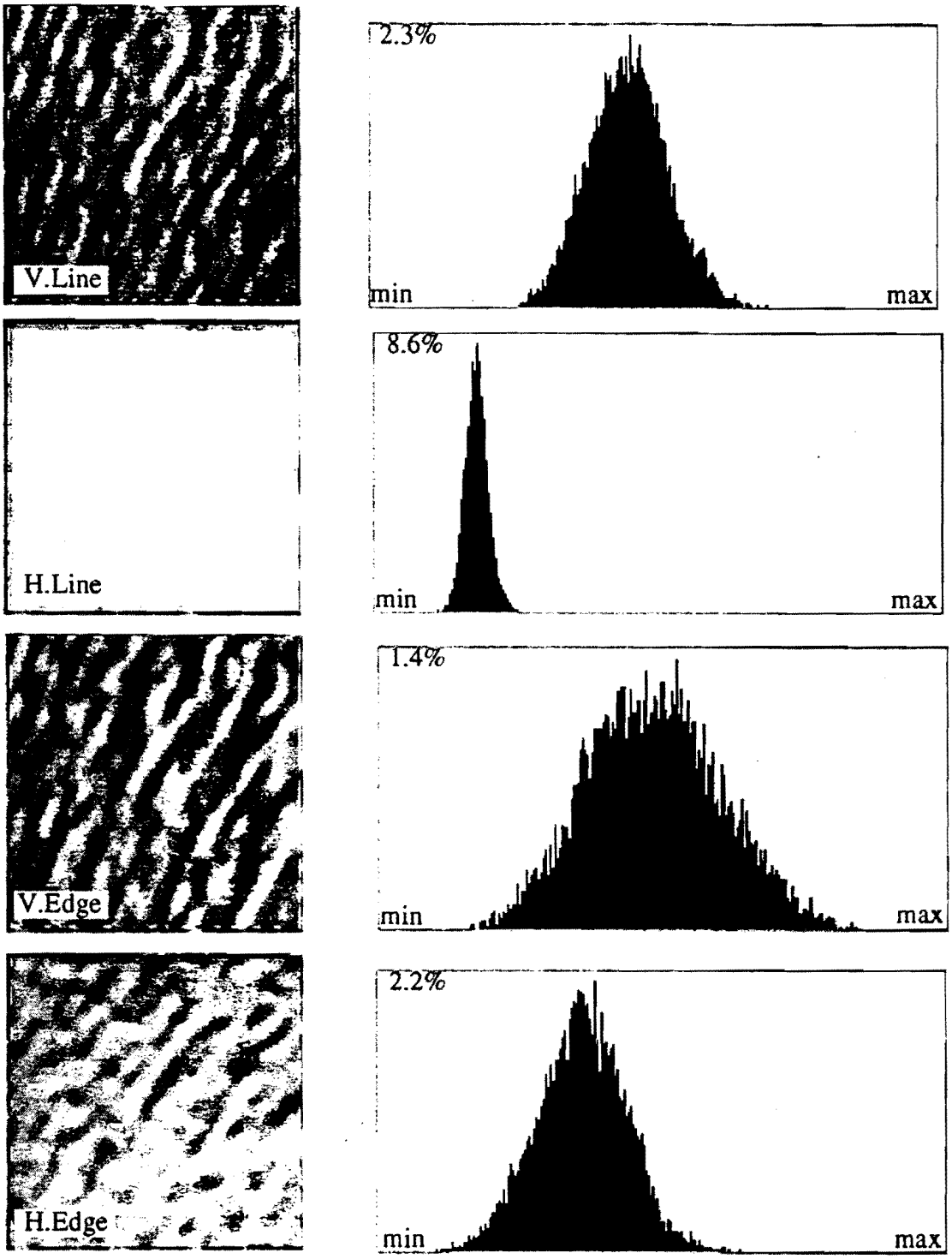


Figure 4-7: Convolved and smoothed areas and their corresponding histograms

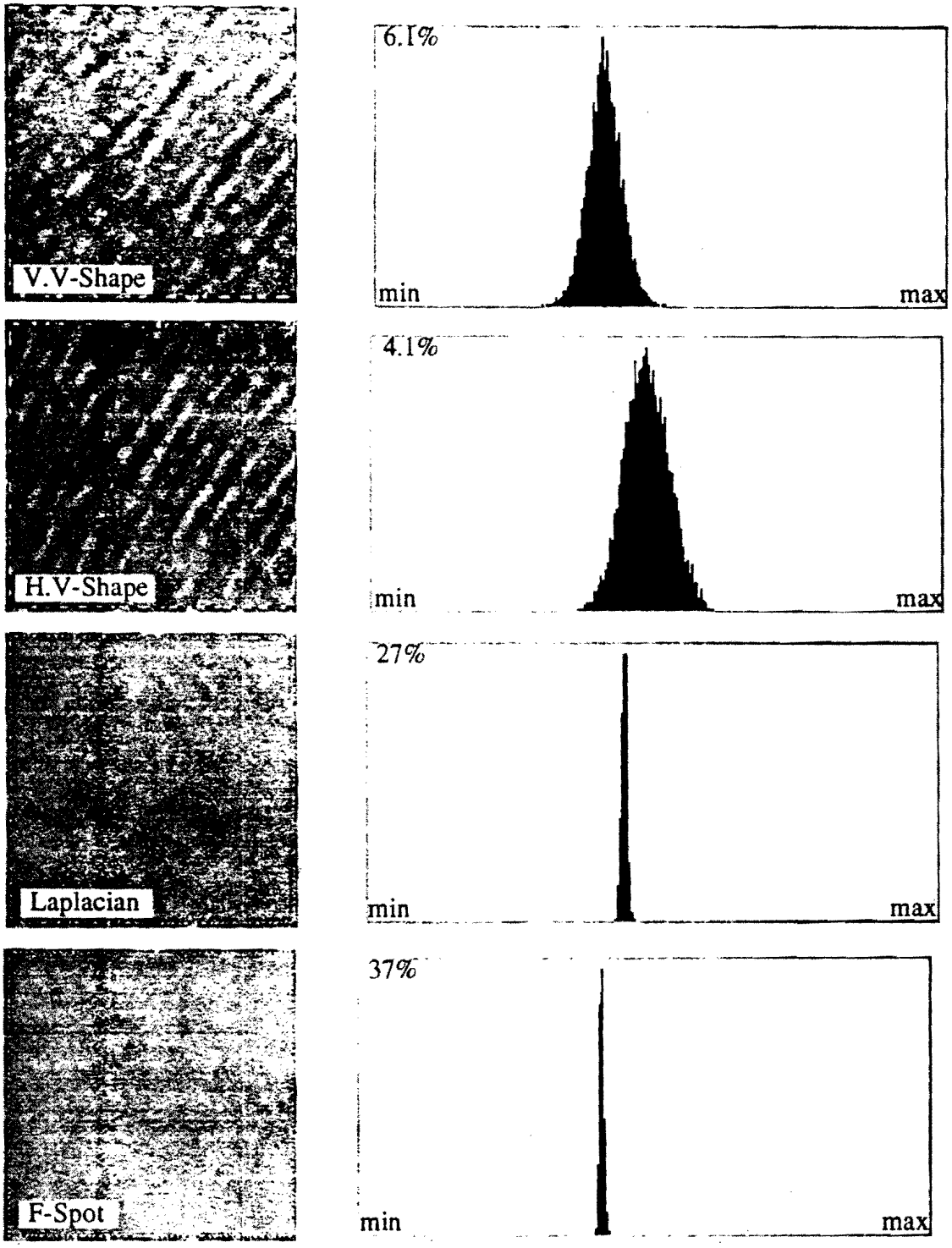
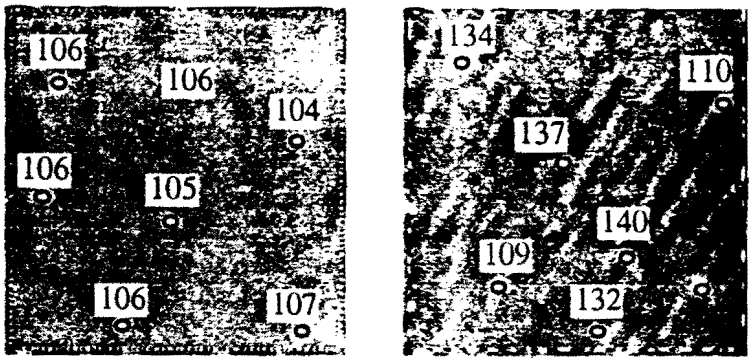


Figure 4-8: Convolved and smoothed areas and their corresponding histograms



F-Spot attribue values H. V-Shape attribute values

Figure 4-9: Initial attributes values in the convolved and smoothed areas

Numeric to symbolic conversion of attribute data can be performed by quantization (scaling) processes. A symbolic attribute corresponds then to an interval of numeric attribute. Pachowicz [Pachowicz, 1989] has already applied different methods of such scaling as an early generalization process of training data. In this work, the simplest scaling was applied [Pachowicz and Bala, 1991], because the number of texture classes is given explicitly and all training examples are computed before the learning step (i.e, concepts are not acquired incrementally). In such a case, the set of scaling parameters $\{\underline{h}min, \underline{h}max, \underline{h}del\}$, where $\underline{h}min=(h\min_1, ...,h\min_8)$, $\underline{h}max=(h\max_1, ...,h\max_8)$, $\underline{h}del=(h\mathrm{del}_1, ...,h\mathrm{del}_8)$, is composed of maxima, minima and lengths of scaling intervals corresponding to each attribute, i.e.:

$$h\min_i = \min \{ Z_i \}$$
$$h\max_i = \max \{ Z_i \}$$
$$h\mathrm{del}_i = (h\max_i - h\min_i)/\#\mathrm{int} \qquad \text{for} \quad i=1,2,...,8$$

where $Z_i = H_i - H_i$ represents selected subset of training events of the i-th attribute. The subset H_i contains training events that are selected from the $H_i = \{h_i(j,k)$: for all (j,k) pixels$\}$ primary set of training data (the set obtained directly after the convolution and averaging operations). H_i contains both a given percentage number ß of training events

from the H_i set that have minimal values, and also the same percentage number ß of training events from the H_i set that have maximal values. Such a selection process creating Z_i attempts to narrow the distance between minimum and maximum values of the scaling process. This selection improves the resolution of an attribute to the most useful range of values. For all experiments presented in this research, the value ß was equal to 2% and the number of intervals #int was equal to 55.

For the above determined scaling parameters $\{\underline{h}min, \underline{h}max, \underline{h}del\}$, the scaling process converts a vector of numeric attributes $\underline{h}(j,k)=(h_1, h_2,..., h_8)$ into a vector of symbolic attributes $\underline{x}=(x_1,x_2,..., x_8)$ in the following way:

```
for m:=1 to 8 do
  if (h_i ≥ hmax_i ) then x_i = #int
  else (if h_i ≤ hmin_i ) then x_i = 0
       else
         for n:=1 to #int
           do
            if (h_i ∈ [hmin_i+(n-1)*hdel_i; hmin_i+n*hdel_i) then x_i=n;
```

4.4 Inductive Learning of Texture Rules

The learning algorithm in the TEXTRAL method is a modified version of the AQ algorithm, developed by Michalski, Reinke, Hong, Mozetic, and others [Reinke, 1984] [Michalski, 1986] [Hong, 1986]. The algorithm is a supervised covering algorithm. The fundamental input to the system are *events*, which are vectors in an n-dimensional attribute space where n is the number of attributes. Attributes can have values of several types. *Linear values* are ordered values such as integers. *Nominal values* are non-ordered, for example, a shape attribute might have nominal values square, circle, triangle. Values may also be *structured* that is, the values of the attribute may form a hierarchy. The shape attribute for example, may also have the value polygon, which could have as sub-values square and triangle, but not circle, which could be a sub-value of the value ellipse.

The *AQ15* program learns decision rules by performing inductive inference on examples and optional initial rules. Training examples are expressed as conjunction of

attribute values, and initial decision rules are logical expressions in disjunctive normal form. The program performs a heuristic search through a space of logical expressions, until it finds a decision rule that is satisfied by all positive examples and by no negative ones and optimized by a rules preference criterion. The program implements the *STAR* method of inductive learning [Michalski, 1983]. Specifically, it is based on the AQ algorithm for solving the general covering problem.

Training examples are given to AQ in the form of *events* . Events belong to decision *classes*. Given a class, events belonging to it represent *positive examples* of the class, and all other events are its *negative examples*. For each class, a *decision rule* or a cover is produced that must be satisfied by all positive examples and by no negative ones.

Each event and decision rule is described by extended selector. An extended selector, or briefly a selector, is a relational statement and is defined as:

[L # R]

where:

L, called the *referee*, denotes an attribute.
R, called the *referent*, is a subset of values from the domain of the attribute L.
is one of the following relational symbols:
=, <, >, >=, <=, <>.

Each rule is assigned two parameters: "t" (for "total weight")—measuring the total number of positive training examples covered by the rule, and "u" (for "unique weight")—measuring the number of positive examples covered by the given rule and not covered by any other rule for the given decision class.

Here is an example of an AQ-15 decision rule:

[Class=1] :: > [x2=1][x4>3][x6=1..7]: (t=6, u=2)

This rule covers 6 examples of Class 1, out of which 2 are covered only by this rule, and not by any other rule for this class. In the case of texture rules, x_i are attributes

characterizing a texture sample (in our experiments we used primarily 8x8 windows). The above rule is satisfied, if attribute x2 takes value 1, attribute x4 has value greater than 3, and attribute x6 takes value between 1 and 7.

Given attributes with values of these types, the algorithm searches through a space of logical expressions relating attributes to values. The goal of the search is to find an expression, which is satisfied by the values for every event in one class, and which is not satisfied by the values of any event in any other class of events. The search is limited by a heuristic called the *lexicographical* evaluation function (*LEF*) which can be user-specified. The LEF evaluates candidate expressions and sub-expressions according to user-specified criteria. Only a small number of expressions are expanded at each stage in the search. The number of expressions to expand at each stage is also a user-specified parameter. The algorithm can be viewed as a beam search through the hypothesis space associated with the vector space defined by the attributes. Figure 4-10 shows the flowchart of the AQ15 algorithm.

The following is an example of a concept description generated by the AQ algorithms. This concept description is a set of 7 rules describing texture class d24 (Figure 4-1).

d24 classs ==>

```
#  cpx
    1   [x1=34..54] [x2=34..54] [x3=4..15] [x4=24,26,28..54] [x5=30..31,33..54]
        [x6=24..45,50] [x7=37..54] [x8=9..17,19..20]  (total:145, unique:145)
or
    2   [x1=29..33,35..39,44,46] [x2=28..34,37..38,41,49] [x3=3..7,9..13]
        [x4=29,32,34,37..39,41..43,46,48,50,53..54] [x5=39..41,43,45..49,52..54]
        [x6=24,26,28..30,32,34..35,38,40,42,48] [x7=42..46,49..52,54] [x8=7..15]
        (total:19, unique:19)
or
    3   [x1=39,43..44,46,48,51,54] [x2=38,40,42,44..46,48,54] [x3=5,7..14]
        [x4=19..21,23,26,31] [x5=32,34,38,48..49,52,54] [x6=31..32,34..35,40..41]
        [x7=26,29,31,34..35] [x8=11..13,15..17,19,21]  (total:11, unique:11)
or
    4   [x1=27,29,32,34] [x2=24,26,29,31,37] [x3=3,8] [x4=23,31,33..34,37]
        [x5=29..30] [x6=20,23..24,28..29] [x7=29,37,39,43,46] [x8=5..6,10]
        (total:5, unique:5)
or
    5   [x1=30] [x2=26] [x3=11] [x4=54] [x5=53] [x6=28] [x7=54]
        [x8=12]  (total:1, unique:1)
```

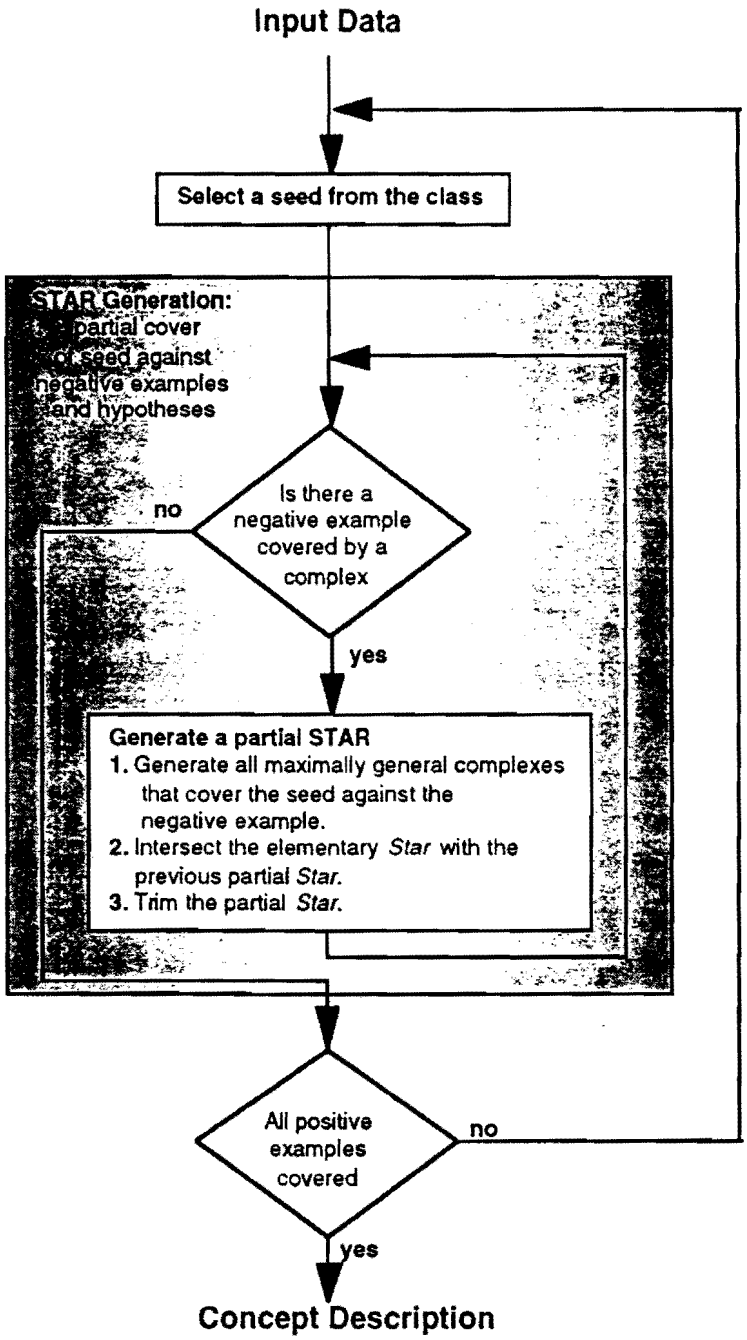


Figure 4-10: AQ15 algorithm

Beside events, declarations of variables, and possibly input hypotheses, AQ15 accepts parameters that specify how the rule should be constructed. The user may select preference criteria according to which the rules are optimized. The criteria measure the quality of the rules from the viewpoint of the specific problem under consideration.

4.5 Rule Application

The learned texture descriptions are generalizations of the observed texture events (i.e., attribute-value vectors characterizing window-size texture samples). Therefore, they can be used to classify unobserved texture samples. There are two methods for applying the descriptions for recognizing the class membership of an event: the *strict* match and the *flexible* match.

4.5.1 Strict Matching

Strict matching is the simplest and fastest matching technique of test data with concept descriptions. In the strict match, one tests whether an example strictly satisfies the conditional part of a rule. Such matching gives the response of two possible logical values, i.e., True or False. Strict matching assumes that concept prototype does or does not cover a given testing examples. If typicality of matched rule is not considered then confidence level of such match can take only two values, i.e., c=1 (an instance matched) or c=0 (an instance not matched). There is no consideration of the degree of closeness between an instance and conditional part of a rule when an instance is not covered by a concept. Therefore, a strict match is not suitable for the classification of noisy data that is usually displaced from clusters of training data.

4.5.2 Flexible Matching

If one assumes that the noisy training data does not truly represent the variability of texture characteristics then one has to accept that testing data can be shifted slightly from main clusters of training data in the attribute space. Additionally, if the recognition phase is preceded by an optimization step then the applied reduction of concept descriptions results in a description that does not even strictly match the entire set of

training data. Thus, to evaluate the membership of any training example one has to apply a flexible match. The flexible match procedure measures the degree of closeness between an unknown example and the conditional part of a rule.

The calculation of the degree of closeness is executed according to the following schema. For a given condition of a rule [$x_n = val_j$] and an instance where $x_n = val_k$, the normalized value of a match of the instance to the conditional part of the rule is computed as:

$$1 - (| val_j - val_k | / \#levels)$$

where #levels is the total number of attribute values. A match to a rule is computed by multiplying evaluation values of matches to each condition of the rule. The total evaluation of class membership of a given test instance to a concept description (set of rules) is equal to the value of the best matching rule. For example, the match c of a test instance:

$x = < 4, 5, 24, 34, 0, 12, 6, 25 >$ (with 55 levels of value each attribute can take)

to a rule:

$$[x_1 = 0] [x_2 = 1..2] [x_7 = 10] [x_8 = 10..20]$$

is computed as follows:

$$c_{x1} = 1 - (|0 - 4| / 55) = 0.928$$
$$c_{x2} = 1 - (|2 - 5| / 55) = 0.946$$
$$c_{x7} = 1 - (|10 - 6| / 55) = 0.928$$
$$c_{x8} = 1 - (|20 - 25| / 55) = 0.91$$
$$c_{x3}, c_{x4}, c_{x5}, c_{x6} = 1$$

$$c = c_{x1} * c_{x2} * c_{x3} * c_{x4} * c_{x5} * c_{x6} * c_{x7} * c_{x8} = 0.74$$

An example of a matching function is depicted in Figure 4-11 (a). This function shows the degree of match for a simple two attributes rule: [x1=2..4] & [x2=3..5] for all points of the representation space (x1, x2). There are 10 levels of values for each

attribute. Figure 4-11 (b) shows four types of regions in the representation space created by the evaluation function. Region type 1 is strictly matched by the rule. Region type 2 represents a part of the representation space satisfied by the selector x1. Points of this regions are evaluated to $1-D_{x2}/N_{x2}$ where D_{x2} is a distance measure from a given point to the rule border lines (border line x2=3 or x2=5) and N_{x2} is the total number of the x2 attribute values. Region type 3 represents a part of the representation space satisfied by the selector x2. Points of this regions are evaluated to $1-D_{x1}/N_{x1}$ where D_{x1} is a distance measure from a given point to the rule border line (border line x1=2 or x2=4) and N_{x1} is the total number of the x1 attribute values. The last region (type 4) represents the areas that are not satisfied by either selectors of the rule. Points of this regions are evaluated to $(1-D_{x1}/N_{x1}) * (1-D_{x2}/N_{x2})$. Figure 4-11 (b) shows that although logic style rules are used to represent class description, the flexible evaluation of a match of unknown example not covered by the rule can be computed. Moreover, depending on the region in which example is located one can detect which selectors (rule conditional parts) are satisfied. This detection introduces some comprehensibility to the flexible matching function.

Classification results for all testing examples are represented as a confusion matrix. The confusion matrix represents information on correct and incorrect classification results. This matrix is obtained by calculating the degree of flexible match between a testing instance and a given class description. The row entries in the matrix represent a percentage of matched instances from a given class (row index) to all class descriptions (column index). Because some of the testing events can be flexibly matched by more than one class, the sum of the entries in a given row may be larger than 100%. Table 4-1 is an example of the confusion matrix for 12 classes.

To recognize an unknown texture sample, the system matches it with all candidate texture descriptions. The assignment of the sample to a given decision class (texture) is based on determining which of the candidate classes gets the majority (or) plurality of votes by the set of examples extracted from unknown texture class. Thus, even if some examples in the sample are incorrectly recognized (non zero values for entries outside the diagonal of the confusion matrix), the classification of the sample may be correct.

Experiments described in this thesis report on recognition accuracy of the first level set of rules. Some experiments with multilevel sets of rules are also presented.

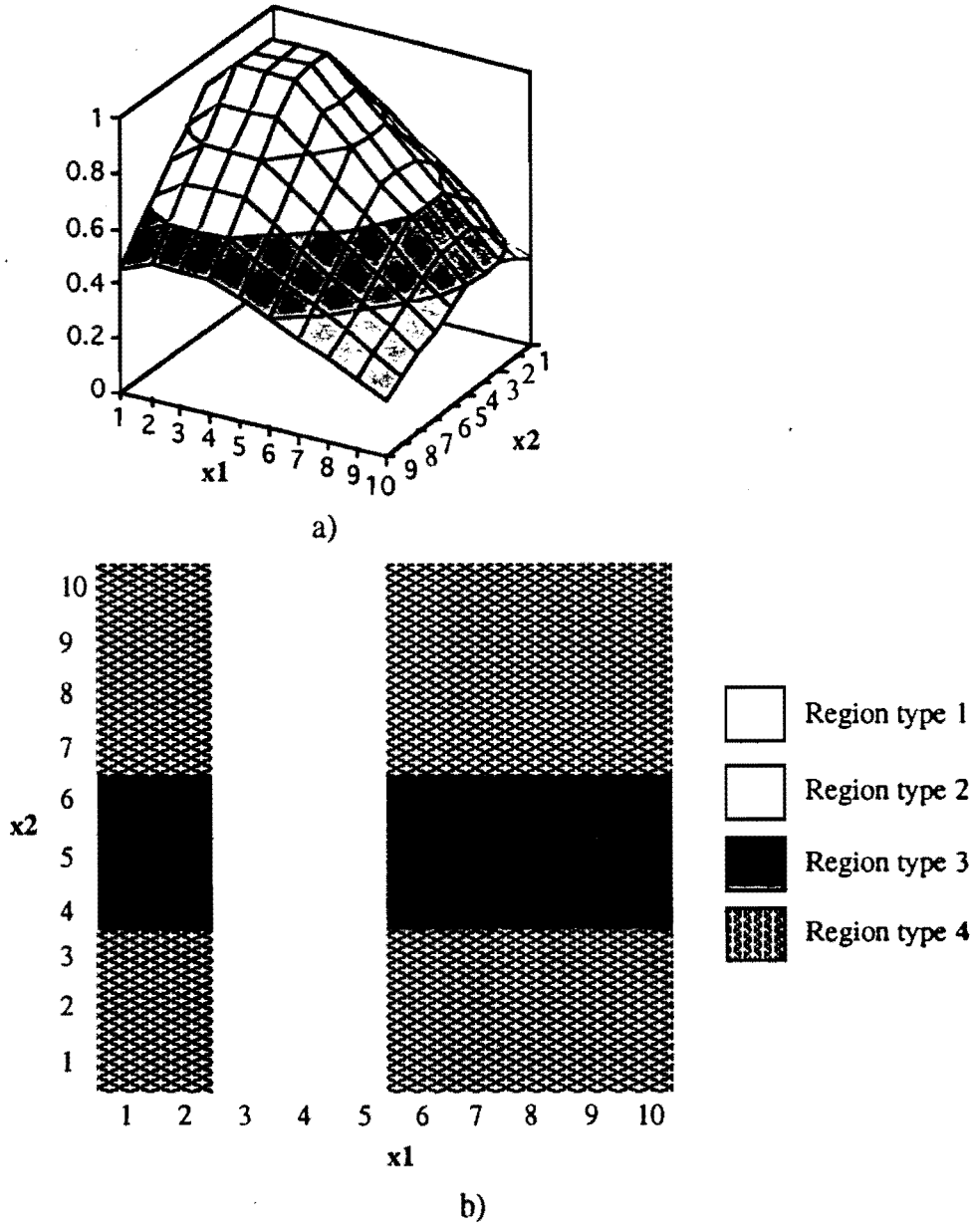


Figure 4-11: Flexible matching function for the rule [x1=2..4] & [x2=3..5], a) a degree of match, b) four types of regions in the representation space

Table 4-1: Confusion matrix for 12 texture classes*

	Test Class											
Learned Class	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12
C1	**45**	5	31	0	0	9	0	28	16	0	6	2
C2	5	**51**	3	12	0	16	1	17	7	0	17	2
C3	31	0	**80**	0	7	1	0	2	10	0	6	5
C4	0	25	0	**72**	0	14	0	12	0	0	1	10
C5	0	0	2	0	**99**	0	0	0	0	0	0	7
C6	8	11	3	25	0	**62**	0	28	0	0	12	1
C7	0	2	0	0	0	0	**98**	0	0	0	0	0
C8	19	14	8	2	1	59	0	**59**	3	0	5	0
C9	16	0	8	0	0	5	0	5	**87**	0	1	2
C10	0	0	1	0	0	0	0	0	2	**95**	0	0
C11	16	4	32	10	0	4	0	4	2	0	**44**	7
C12	0	0	0	4	0	0	1	0	0	0	2	**98**

4.6 Texture Rule Optimization

One of the most important problems in the application of machine learning to vision data is the influence of noise. Image data is corrupted with noise, and consequently, learned concept descriptions contain noisy components. The main objective of optimization methods is to decrease the influence of noise on learned descriptions according to some performance measure.

Most of the inductive learning systems generates concept descriptions according to some performance measures. These performance measures are expressed as the preference criteria that guide inductive search for the best local elements of a concept. A single concept component is derived separately and then integrated with a concept structure. Thus, a single component is locally optimal according to a given preference

* From [Bala, DeJong et al., 1992].

criteria. In the second phase, concept manipulation, the system analyzes sets of concept components and modifies concept descriptions. Criteria for the modification of concept descriptions are different from criteria of component acquisition. These criteria consider concepts performance and overall rather than local structure.

The problem of acquiring well performing concept descriptions has been studied intensely. One of the first approaches incorporated selection of most representative training data and the learning in the incremental mode [Michalski and Larson, 1978]. The problem of manipulation of concept descriptions was then discussed from different perspectives (see for example: [Fisher, 1988], [Iba, 1988], [Markovitch and Scott, 1988], [Tambe, 1988], [Holte, 1989], [Tcheng, 1989], [Weiss and Indurkhya, 1991]. Concept description manipulation techniques were implemented within several learning programs, for example, pruning decision trees was implemented for ID family of programs [Quinlan, 1987], [Quinlan, 1989], and SG-TRUNC optimization method was implemented for the AQ family of programs ([Zhang and Michalski, 1989], [Bergadano, Matwin et al., 1992]). These approaches, however, have common problem --- they learn from original noisy data without trying to improve training data during the learning process.

4.6.1 Optimization Model

This section introduces an optimization model [Pachowicz and Bala, 1991]. The optimization process is applied in order to modify concept descriptions in such a way that optimized descriptions reach a higher quality measure when compared with primary descriptions; i.e.,

$$O:\ D \times Z \xrightarrow{\quad Q \quad} D$$

$$\text{and} \quad Q[\, r(d^*,x)\,] \;=\; \max_{Z} Q[\, r(d_Z,x)\,]$$

where: $d^* \in D$ is optimal concept description according to a given Q quality measure, $d_Z = o(d,z)$ is an optimization process, Z is the space of optimization parameters, $r(d,x)$ is a recognition process, and $x \in X$. is a concept example (X is the representation space).

To follow the consequences of applied concept description optimization, let us study the characteristics of recognition processes under the following assumptions: (i) class descriptions are learned by the AQ method, (ii) training data is noisy, and (iii) hyper-spheres of training examples overlap for different classes through the attribute space. Let us consider the simplest case of class distributions; i.e., there are only two classes, where the first class (Ω_1) has uniform distribution of training data, and the second class (Ω_2) has normal distribution of training data ($N(\mu, \sigma)$). The Ω_1 class represents background noisy training data belonging to other hypothetical classes within attribute space that cause the partitioning of description of the Ω_2 class. Learned descriptions of both classes will contain more than one concept component (rule) because the learning process derives a cover (a rule) over positive examples only and none of negative examples.

Statement 1: For two classes in the attribute space, where one of them (Ω_1) has uniform distribution and the second one (Ω_2) has normal distribution, the relationship between acquired concept components ($rule^{\Omega 2}_k$) of the second class (Ω_2) is as follows:

$$t(\Omega_2,n) > t(\Omega_2,m) \;\Longrightarrow$$
$$P(\rho(rule^{\Omega 2}_n, \mu) < \rho(rule^{\Omega 2}_m, \mu)) > P(\rho(rule^{\Omega 2}_n, \mu) > \rho(rule^{\Omega 2}_m, \mu))$$

where,
$t(\Omega_2,n)$ and $t(\Omega_2,m)$ are typicality measures of n, m components (rules) of the Ω_2 class description , and $\rho(rule^{\Omega 2}_n, \mu)$ is the distance from the center of $rule^{\Omega 2}_n$ to to μ (the center of Ω_2).

The presented statement means that more typical concept components (i.e., covering more training examples) are generally closer to the center of a cluster of training data than less typical concept components (the total-weight can be used as the measure of typicality of concept description components in the AQ method, Section 3.1). In this way, concept component typicality depends on the distance measure from the center of local cluster of training data to a concept component. If the area of concept components of class Ω_2 is similar then the distribution of typicality measure $t(\Omega_2,j)$ is a normal-like distribution regarding the center of data cluster. The assumption of similar area of

concept components of class Ω_2 is generally fulfilled by the distribution of examples belonging to class Ω_1 that partition the description of class Ω_2.

Statement 2: For more than two overlapping classes in the attribute space, where one of them (Ω_1) has uniform distribution and the other classes (Ω_2, Ω_3, ..) have normal distributions, the distribution of $t(\Omega_k,j)$ typicality measure for $k=2,3,...,n$ is not a normal-like distribution at the boarder area between classes.

The irregular distribution of typicality measure is caused by the overlapping effect of two (or more) classes (Figure 4-12). It causes higher partitioning of concept descriptions within the border area between two classes if they are overlapped. The effect of a higher partitioning on concept descriptions is certainly negative. It means that concept partitioning increases with an increase in the standard deviation of the distribution of training data. The increase in variance causes two classes to overlap with higher degree, i.e., the border line between both classes tends to disappear. The partitioning problem is much greater when the distance between cluster centers of two classes decreases. While the traditional pattern recognition method of Bayes risk minimization is able to approximate a cross-section between two overlapping classes, the introduced machine learning method is affected by the mentioned problems. The negative effect of description partitioning causes a single instance to be classified incorrectly (i.e., matched with noisy concept component of any counterclass) even if it is in the center of its class membership. This situation occurs when a noisy concept component is derived across the border with an other class description (i.e., on the opposite site of the valley). This component can then be matched with test data incorrectly. The partitioning problem has been ignored by most previous research. Recently, [Whitehall, Lu et al., 1990] developed the CAQ algorithm that searches for the border between distributions of numeric attributes of different classes. Their approach, however, assumes that the distribution of training data is known explicitly.

The effect of partitioning on concept descriptions can be utilized through the optimization of concept descriptions to increase system recognition effectiveness.

Theorem 1: If $D^* \subset D$, $d = < d^1, d^2, \ldots, d^{\#classes} > \in D$, and $d^* = o(d) \in D$ is optimized in such a way that if the number of less significant concept

components is reduced then the recognition process r(d*,x) matching an instance x with optimized description d* can perform better than the recognition process r(d,x) matching the same instance x with primary description d.

Proof:

Removing less significant concept components from the description d decreases the effect of class overlapping.

(i) the probability that the less significant concept component of a given Ω_i class is eliminated from the area of clusters typical for counter-classes is higher than from the area occupied by clusters of that class itself

(ii) removal of less significant concept components decreases negative partitioning of attribute space, and

(iii) the decrease in partitioning of attribute space clears both the area of cluster centers and the border areas between class descriptions.

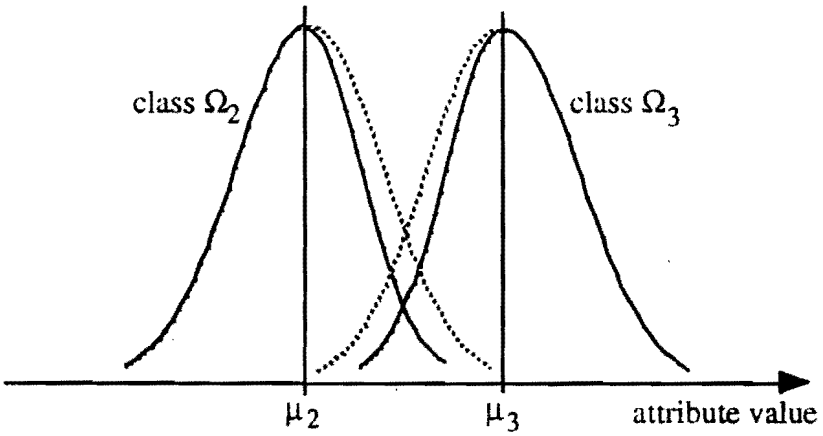


Figure 4-12: Irregular distribution of typicality measure

The optimization of a description d produces description d* that has clearer cluster areas and clearer border areas. The cluster area of one class has less noisy concept components than other classes. The border areas between class descriptions are more

distinct allowing them to match test instances with closer and more significant concept components. Thus, the probability that test instance x located within cluster areas of its class membership is classified correctly, is higher for optimized description d* than for the primary description d.

The degree of concept optimization plays a major role in a manipulation of concept descriptions. The increase in optimization parameters is expected to be followed by an increase in a quality value representing system performance because the system will no longer match test instances with less significant concept components. Substantial increases in optimization parameters, however, will have a negative effect on system performance. Such an increase can cause the removal of more significant concept components, especially when the distribution of such a concept is very irregular (e.g., when the training data has many local clusters or it forms a complex path through the attribute space).

4.6.2 Optimization Criteria and Quality Measures

The quality of a segmented and annotated image depends on the quality of all three phases of the texture recognition and segmentation schema, i.e., (i) image processing performed to extract texture attributes, (ii) matching image elements with learned texture class descriptions in order to annotate them by most probable classification hypotheses, and (iii) local unification of classification hypotheses in order to segment an image into homogenous areas corresponding to certain objects. Quality criteria for the evaluation of segmentation processes are precise [Zucker, Rosenfeld et al., 1975], [Davis, Clearman et al., 1981], [Du Buf, Kardan et al., 1990]. A perfect system should: (i) preserve sharp and precise borders between different texture areas, (ii) smooth homogeneous texture surface areas, and (iii) preserve small objects against their removal from the segmented image. These criterion are still difficult for current computer vision systems. The criterion of smoothing texture surface areas requires the extension of a radius of local operators extracting texture attributes. On the other hand, if a radius is enlarged then the operators blur borders between texture areas and they can remove small objects from an image.

Considering the hierarchical processes of texture recognition and segmentation schema and their mutual dependencies caused by the quality of their performance, the following performance criteria have been selected for texture recognition system [Pachowicz and Bala, 1991]:

- increase the classification accuracy when matching a class description with examples of this class (*maximum accuracy criterion*),
- decrease the classification accuracy when matching examples with other class descriptions (*minimum misclssification criterion*), and
- perform on similar classification accuracy level for all classes when examples are matched with their class descriptions (*system stability criterion*).

Considering the above requirements, a system should search for high recognition rate (recognition accuracy) and low misclassification rate. Moreover, the system stability criterion requires that each concept should have the same chance to be recognized. Highly negative effect is reached when one class can be recognized with the highest confidence (e.g., above 95%) and the other class with relatively low confidence (e.g., below 60%).

Experiments presented in Section 5.3 (next chapter) were evaluated by testing optimized concept descriptions on separated sets of test data. For each subset of 200 examples randomly extracted from a given homogenous texture area representing i-th class (different than the area used for the extraction of learning examples), the system computes a recognition rate (accuracy) for this class. The recognition rate for class Ωi was calculated by dividing the number of correctly classified test events from the $Tevents^{\Omega i}$ test dataset by the total number of test events for this class; i.e.,

$$rec\_rate^{\Omega i} = \frac{\#\{\, x:\ x \in Tevents^{\Omega i} \text{ and } r(d,x) = <\Omega i_{\text{matched}}> \,\}}{\#\{\, x:\ x \in Tevents^{\Omega i} \,\}}$$

System recognition effectiveness was then evaluated through the computation and monitoring of the following measures:

1) average recognition rate computed through all twelve test datasets representing texture classes,
2) standard deviation from the average recognition rate (system stability criterion prefers the standard deviation to be minimum), and
3) minimum recognition rate representing the worst performing concept description (this rate should be as high as possible to allow all classes to be recognized).

4.6.3 Concept Description Optimization Methods

Two classes of concept description optimization methods can be distinguished ; i.e., direct and indirect. Direct optimization methods (Figure 4-13) directly manipulate acquired concept descriptions. This manipulation involves both specialization and generalization operations. Specialization operations shrink a concept through elimination of concept less significant components. On the other hand, generalization extends a concept description over nearest and hypothetically noisy (less significant) components belonging to counterclasses. The second approach to concept description optimization, indirect optimization, incorporates pre-optimized concept descriptions (i.e., optimized by a direct method) to filtrate the final training data [Pachowicz and Bala, 1991]. The learning process is then repeated using modified (truncated) set of training data. This method is presented in chapter 6.

4.6.3.1 Simple Truncation

The first direct optimization method of concept descriptions is based on the theory of Two-Tiered Representation (TT) of flexible concepts [Michalski, 1987]. The theory assumes that an acquired concept description can be transformed to its TT representation through a separation of the most significant concept properties (Base Concept Representation) from exceptions to these properties (Inferential Concept Interpretation). Since the concept descriptions learned by the family of the AQ programs are composed of ordered components (i.e., from the most to less significant components), one can truncate such descriptions by removing some less significant components. In experiments (Section 5.3), the truncation degree is controlled by a parameter corresponding to the

percentage number of training examples covered by removed components to all training examples.

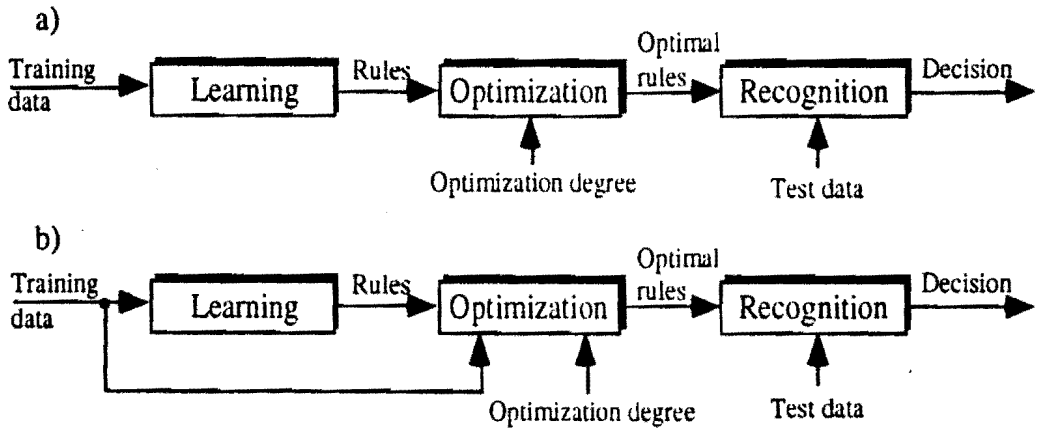


Figure 4-13: Direct optimization methods

4.6.3.2 SG-TRUNC Method

The SG-TRUNC method is another direct optimization method applied to improve the performance of concept prototypes [Zhang and Michalski, 1989]. This method incorporates both specialization through the truncation of less significant concept components and also generalization of concept components. The generalization of concept components, however, is performed through the extension of attribute values within concept descriptions. In this way, a concept component covers not only positive training examples but it can cover some negative examples as well. The degree of allowed coverage is controlled by two parameters.

The SG-TRUNC optimization method was implemented within the AQ16 integrated learning system. The optimization process requires training data (Figure 10-13 b). The authors of this method demonstrated its ability to improve system recognition effectiveness for different domains.

4.7 Multilevel Learning in TEXTRAL

As was described in chapter 3, the TEXTRAL method incorporates an iterative (multilevel) application of symbolic inductive learning to generate texture rules. In this mode, a learning process is repeated iteratively until the desired textural area transformation is obtained. The transformation of textural area is accomplished by matching examples extracted from all pixel position in the learning area with description learned in the previous iteration. The results of this matching is a class membership decision obtained for each example. Since each example is extracted from some position in the learning texture area, this position is labeled with class name. The new regenerated learning area of the given texture now represents a symbolic image. This symbolic area shows how well a texture class was segmented by its learned description. In all the following iterations, the system learns from segmented areas. In each iteration, average recognition accuracy is monitored.

The method consists of two phases: one that extracts information from raw textural images by applying convolution operators and learns an initial set of rules; and a second that iteratively extracts symbolic information from the transformed representation of initial image and learns another set of rules. The first phase represents an application of Law's masks operators to learn the first set of rules. The second phase is the iterative process of symbolic image regeneration (based on rules learned in the previous iteration step) and the learning of a new set of rules by extracting symbolic events and applying the AQ inductive method. In the second phase, a new operator, different than in the first phase, is used to extract example from symbolic image.

The extracted examples from each texture class are input into the AQ learning module. Generated descriptions (rule sets) describe discriminatory properties of a given set of classes and are stored in a *D[1..k][1]* (with i index equal 1, Figure 4-14). These descriptions form the first column of the *D[][]* matrix. They are used to generate symbolic representations textural learning areas. The same extraction process is repeated, but this time to different pixel positions of the learning area. Each extracted event is evaluated by flexibly matching with *D[1..k][1]* (descriptions learned from the initial extraction process). The result of this evaluation is used to determine the class membership of a

given extracted examples. The determined class name (symbolic value) is used as a symbolic pixel value in a transformed symbolic representation of a given textural area. Each pixel position in the learning area is assigned its symbolic value based on the dominant symbolic value assigned by the evaluation process in the neighborhood of this position. To accomplish this, a small window (e.g., 5 by 5 pixel positions) is scanned through the learning area, and the central pixel position in this window is substituted by the dominant symbolic value inside the 5 by 5 area. This operation of *symbolic image smoothing* is necessary since not all pixel positions of the learning area are extracted for evaluation. This symbolic image represents a classification (or mis-classification) of the spatial characteristics of the first extraction and learning processes (for i=1).

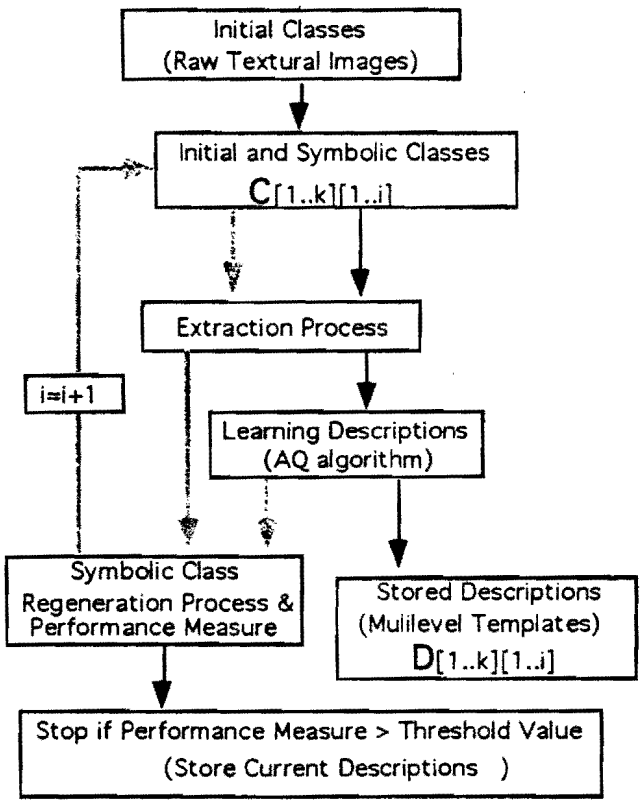


Figure 4-14: Multilevel learning *

* k is the class index, i is the iteration index, shaded lines depict the regeneration process.

A different extraction process (than the one used for the initial image) is used to derive example sets for the next iteration of the signature learning algorithm. The extraction operator is a simple window of different sizes (5 by 5 was usually used in most experiments) which extracts symbolic values of neighboring pixels *(x1,x2,..,x8)* as depicted in a Figure 4-15.

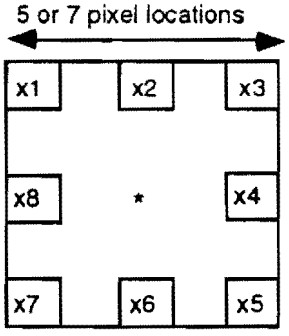


Figure 4-15: Extraction operator for symbolic image

This operator is used in all subsequent learning and regeneration processes. Let us suppose that we have four classes *(A,B,C,D)*. The extracted symbolic event might look like this:

$$<x1, x2, x3, x4, x5, x6, x7, x8> = <A, B, B, C, D, D, A, D>$$

As we can see the extracted volume of information from symbolic representation is substantially reduced.

Each next iteration adds one column to the D matrix. A row in a matrix represents a sequence of descriptions. This sequence of descriptions together with extraction operators is called a *texture signature* (Chapter 3). Each description can be optimized as explained in the previous sections.

After each new symbolic image regeneration process is completed statistics for all classes in form of a confusion matrix are computed. Using confusion matrix the method estimates how well each class is represented by pixels with this class name value. A threshold is introduced to determine how well a given class should be represented in a transformed form by its name in a symbolic image. Let us say, if more than 90% of pixel location of a given class are assigned (by the matching process) its name we do not proceed with the next iteration step for this class. If more than 90% of the pixels are properly assigned, we say that this class is strongly represented by the descriptions derived from the previous extraction process. This also means that a given extraction process is relevant to the discriminatory characteristic of a given class. If less than 90% of pixel locations are assigned the name of the class, the iteration process is repeated. If a class is mis-represented in its symbolic representation (majority of pixels are assigned other class name) the next step of the learning algorithm must proceed (probably yielding correct classification results in the next regeneration process). The ability to "force" the correct classification results by generating the next rule description of a given class (next step of regeneration and learning processes) is an important and novel feature of this method. This feature provides immunity to the extraction method chosen.

The description matrix D is used to recognize unknown textural areas. The same sequence of extraction operators is used as describe previously (first convolution operators, followed by extraction window, as in Figure 4-15). The extraction events are flexibly matched against the first column of signature matrix. Based on matching results, the unknown textural image is transformed into its symbolic representation. The extraction process is repeated, but this time using a symbolic leve extraction operator (Figure 4-15). During each iteration step there may be fewer classes to be matched with the still unknown class. The determination of class membership can be made during each iteration step depending on the matching results. If the unknown class is matched strongly (above the threshold value) to some class, the next iteration of the recognition algorithm is not needed.

Chapter 5

5. Experimental Results

5.1 Introductory Experiments

This section presents introductory results of applying a learning approach to the acquisition of texture class descriptions [Pachowicz and Bala, 1991]. The experiments show the dependency of learning and recognition processes on variable learning conditions. The following conditions are tested:

1. the size of texture feature extraction window for the computation of local macro-statistics (explained in Section 4.3),
2. the number of training examples, and
3. the acquisition of specific and general concept descriptions.

All experiments used 12 classes of texture presented in Figure 4-1. The extraction of texture attributes, learning processes, and flexible matching processes were performed as explained in Chapter 4.

5.1.1 Extraction Parameters

The effectiveness of a learning schema applied to the acquisition and recognition of texture concepts depends both on the processes preceding the learning phase and the parameters of learning program. These processes are related to the extraction of texture attributes and the selection of training data. The following parameters were investigated:

(1) the size of averaging window applied to compute local macro-statistics of texture energy (Laws', 1980), and

(2) the number of training examples provided for the learning phase.

Acquired characteristics of system recognition effectiveness are presented in Figure 5-1. In this experiment the training dataset was adjusted from 50 learning examples per class to 300 learning examples per class. The test dataset was constant and it consisted of 200 testing examples per class. The radius of the averaging window applied to acquire macro-statistics of texture energy (Figure 4-5) was 3.5, 5.5 and 7.5 pixels.

Figure 5-1 shows that the average recognition rate increased slightly with an increase in the number of training examples. At the same time, the standard deviation decreased and the minimum recognition rate increased rapidly for larger window size. This observation indicates that an increase in the number of training examples has substantial influence on system recognition stability. On the other hand, overall system recognition effectiveness, measured by the average recognition rate, is very sensitive to the window size. The overall effectiveness improved substantially with larger windows.

Based on this evidence, one could consider an increase both in the window size and the number of training examples in order to improve system recognition effectiveness. An increase in window size, however, has significant influence on the image segmentation processes [Tomita and Tsuji, 1977] and must be limited to the area dependent on the content of texture image (i.e., size and shape of texture areas). A significant increase in the amount of training data is impractical because of the increase in concept complexity.

Considering the results of applying a learning approach to the acquisition of texture concepts from noisy texture data, one finds that the improvement of system performance must be done in a way other than an adjustment in window size and the increase in the number of training examples. A certain balance is required in order to support high recognition effectiveness, high stabilization of recognition decisions, and relatively low complexity of concept descriptions. In further experiments an extraction

window of radius R=7.5 and the number of training data of 200 examples representing each texture class were used.

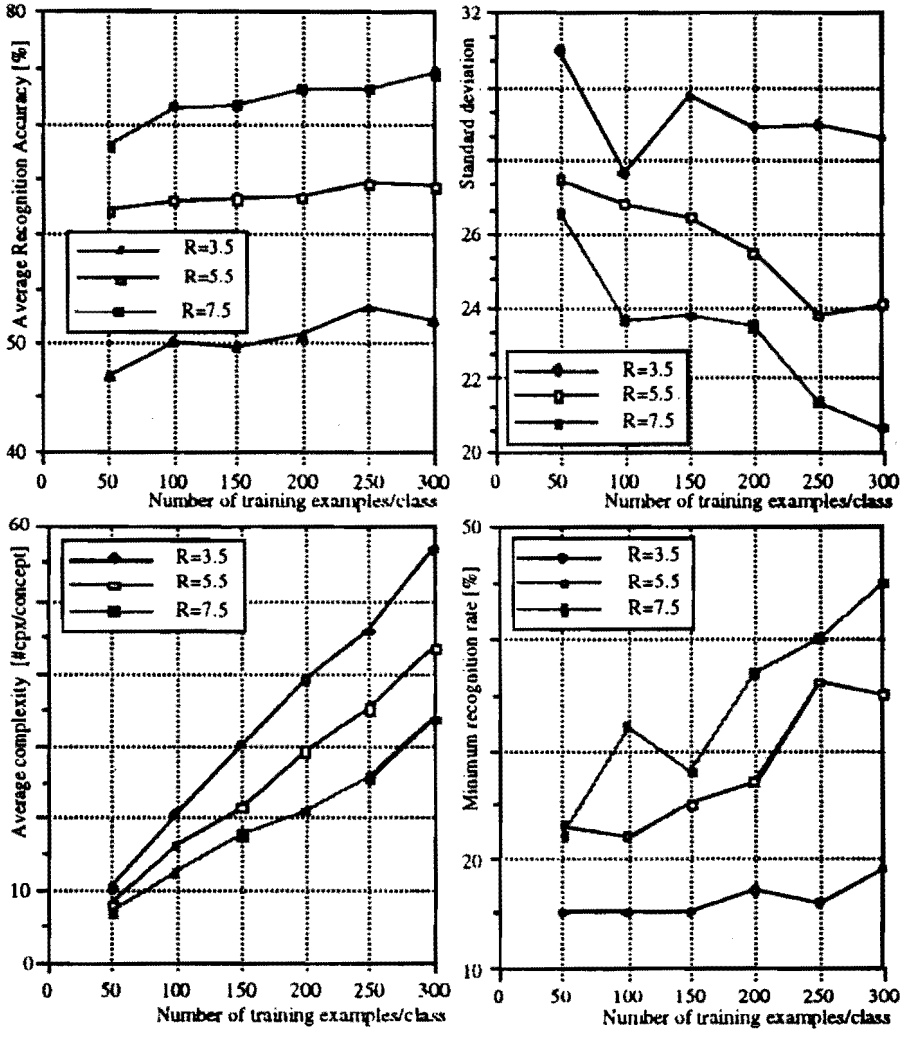


Figure 5-1: Learning characteristics

5.1.2 Specific *Versus* General Concept Descriptions

The difference between specific and general concept descriptions can be illustrated by the area of attribute space separated by these descriptions. General descriptions cover

a larger area than specific descriptions; i.e., the conditional part of a general rule contains larger range of attribute values than a specific rule. Moreover, general descriptions overlap a larger area than specific descriptions. Thus, general descriptions can be matched strictly with test data over larger area of the attribute space. The AQ learning programs can generate both, specific and general concept descriptions, by using a *trim* parameter.

For general description, the probability that an instance is classified to more than one class is higher than for a specific description. The primary effect of such matching is the increase in the average recognition rate — where the recognition rate is not a probability measure and it is computed as the ratio of the number of instances classified correctly to the concept by the total number of test instances. A test instance, however, can be covered by more than one general concept which allows classification to more than one class (see Table 4-1 in previous chapter). In this way, a highly negative effect of such matching is seen as an increase in misclassification rate. The misclassification rate monitors both the number of instances that are classified incorrectly and the number of instances that are not uniquely classified to the correct class.

Figure 5-2 presents the improvement in average recognition rate, standard deviation, and the minimum recognition rate when concept descriptions are learned as general descriptions (black dots) and specific descriptions (white dots). This improvement is illustrated for different sizes of the attribute extraction window and for different numbers of training examples. These results suggest it is better to learn general rather than specific concept descriptions. The misclassification rate, however, suggests otherwise. An example is shown in Table 5-1 and 5-2, where confusion matrices are presented both for general and specific concept descriptions. The average misclassification rate shows a nearly two fold increase when general descriptions are applied to recognize test data. The minimization of such misclassification rate is important for the image segmentation phase. Thus, the choice of general or specific concept descriptions must consider both the average recognition rate and the average misclassification rate. Considering the large difference in the misclassification rate, further experiments will be based on the acquisition of specific rather than general concept descriptions.

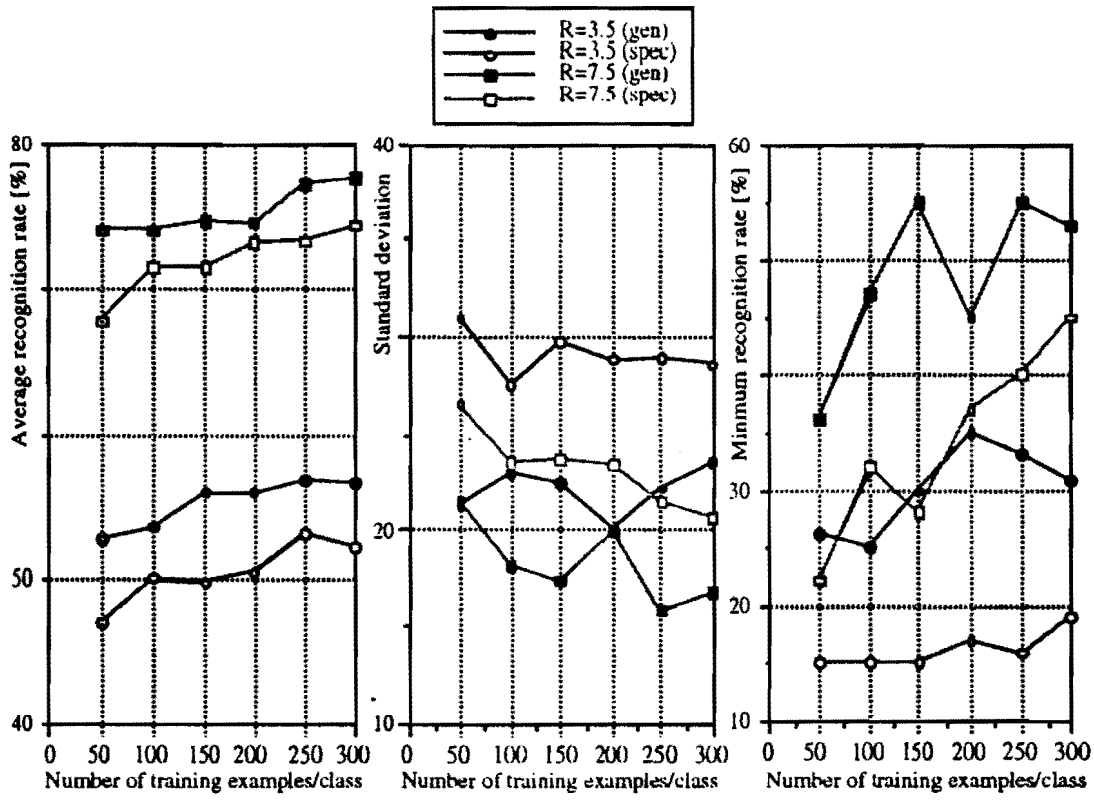


Figure 5-2: Recognition effectiveness of specific and general descriptions

Table 5-1: Confusion matrix for general concept description

	d4	d5	d9	d19	d24	d28	d37	d54	d57	d77	d92	d93
d4	**45**	5	31	0	0	9	0	28	16	0	6	2
d5	5	**51**	3	12	0	16	1	17	7	0	17	2
d9	31	0	**80**	0	7	1	0	2	10	0	6	5
d19	0	25	0	**72**	0	14	0	12	0	0	1	10
d24	0	0	2	0	**99**	0	0	0	0	0	0	7
d28	8	11	3	25	0	**62**	0	28	0	0	12	1
d37	0	2	0	0	0	0	**98**	0	0	0	0	0
d54	19	14	8	2	1	59	0	**59**	3	0	5	0
d57	16	0	8	0	0	5	0	5	**87**	0	1	2
d77	0	0	1	0	0	0	0	0	2	**95**	0	0
d92	16	4	32	10	0	4	0	4	2	0	**50**	7
d93	0	0	0	4	0	0	1	0	0	0	2	**98**

Table 5-2: Confusion matrix for specific concept description

	d4	d5	d9	d19	d24	d28	d37	d54	d57	d77	d92	d93
d4	**43**	2	17	0	0	1	0	27	12	2	0	0
d5	4	**54**	2	8	0	8	0	10	9	0	6	1
d9	15	0	**66**	0	5	0	0	0	8	0	1	7
d19	0	7	0	**88**	0	5	0	1	0	0	1	5
d24	0	0	0	0	**99**	0	0	0	0	0	0	0
d28	10	10	2	20	0	**45**	0	15	0	0	6	4
d37	0	0	0	0	0	0	**99**	0	0	0	0	0
d54	16	5	4	1	0	8	0	**61**	0	0	4	0
d57	3	0	8	0	0	0	0	1	**88**	1	0	0
d77	0	0	0	0	0	0	0	0	0	**99**	0	0
d92	11	5	30	4	0	7	0	2	0	0	**37**	7
d93	0	0	0	1	2	0	0	0	0	0	0	**97**

5.2 Experiments with Multilevel Learning

Experiments presented in this section were performed using the iterative mode of the TEXTRAL method [Bala and Michalski, 1991].

Tables 5-3 and 5-4 show the confusion matrices characterizing the system's recognition rates (in %) for individual texture events selected from testing areas of four texture classes, C1, C2, C3 and C4. Table 5-3 shows the recognition rate for first level rules, and Table 5-5 — for the second level rules. Recall that the conditions of the second level rules apply not to properties of the original image, but to the distribution of texture labels generated by the first level rules.

The initial texture classes and their symbolic representations for the first and the second iterations of the learning algorithm are shown in Figure 5-3.

Table 5-3: Recognition rates using the first level rules

	Recognized texture class			
Correct Class	C 1	C 2	C 3	C 4
C 1	*84*	*15*	*16*	*23*
C 2	*10*	*78*	*20*	*10*
C 3	*7*	*14*	*79*	*27*
C 4	*26*	*17*	*27*	*67*

Table 5-4: Recognition rates using the second level rules

	Recognized texture class			
Correct Class	C 1	C 2	C 3	C 4
C 1	*94*	*3*	*2*	*6*
C 2	*4*	*96*	*4*	*4*
C 3	*4*	*9*	*88*	*12*
C 4	*13*	*7*	*12*	*80*

The average correct recognition rate of individual events for the 4 class experiment was 77% when using the first level rules, and 89.5% when using the second level rules. At the same time, the average misclassification rate decreased from 17.6% to 6.6%, respectively. Thus, the experiment has demonstrated that multilevel learning (using higher level rules) can increase the system's recognition of individual events. The learning area for each class was chosen to be a 100 by 100 square. From each class 100 events were derived to learn the rules in each iteration step. For each regeneration process 500 events were randomly chosen inside the learning areas. After completing the matching process, a 5 by 5 window was scanned through all pixel positions (100 by 100) of the learning area and the dominant recognized class inside this window was used as the symbolic value of the pixel represented by the central position inside this window (see section 5). By using this technique all 100 by 100 pixel positions were assigned their

symbolic value (although only 500 had been used for regeneration/matching process). The same number of learning and regeneration events, and the technique of a dominant class inside a 5 by 5 window, were used in other experiments.

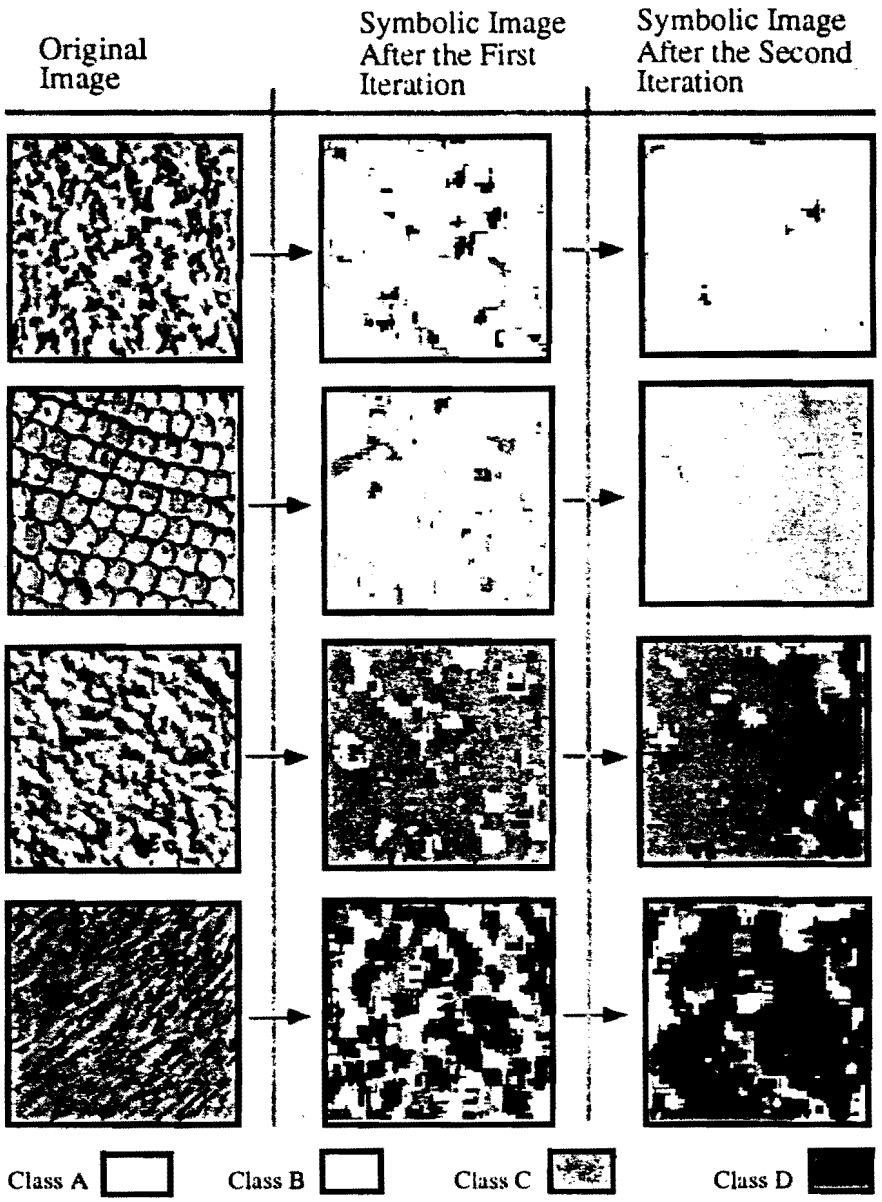


Figure 5-3: Four texture samples and their symbolic representations

Figure 5-4 shows the results of the 8 classes experiment. The following average recognition accuracy are obtained from each iteration (averages of entries in the diagonal of three confusion matrices): 62.6%, 77.3%, 79%. This is an increase of 16.4% of the correct recognition rates between the first and the third iteration. The recognition rate of the class recognized increased from 53 % to 60 %. The average misclassification rate decreases from 32.1% to 8%.

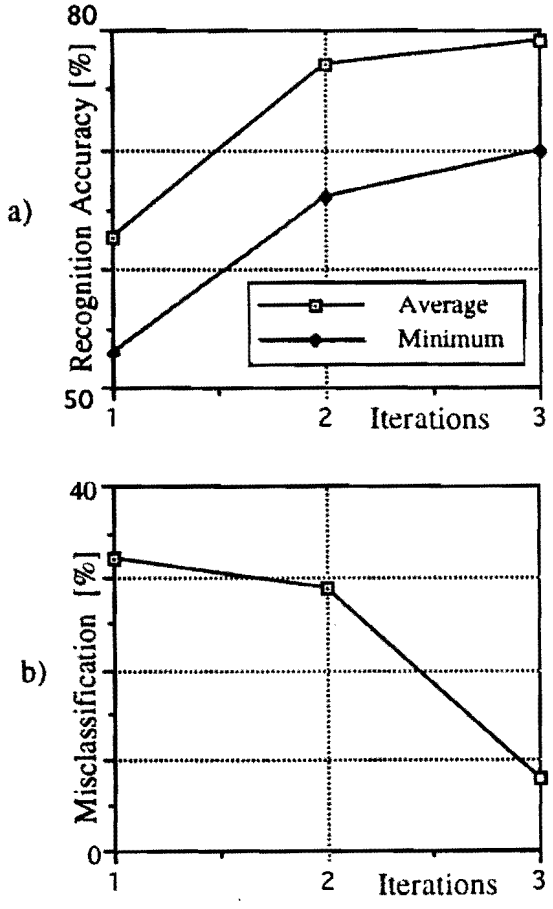


Figure 5-5: The eight textures experiment

Figure 5-5 presents the results of the 12 textures experiments (textures in Figure 7). Each class was correctly recognized in each of four iteration steps. All classes were correctly recognized in each iteration step. The average recognition rate increased from 48% to 58%. At the same time, the standard deviation decreased from above 20 to 15. Minimum recognition rate increased from 21% to 36%. All these significant changes were obtained in only four iterations.

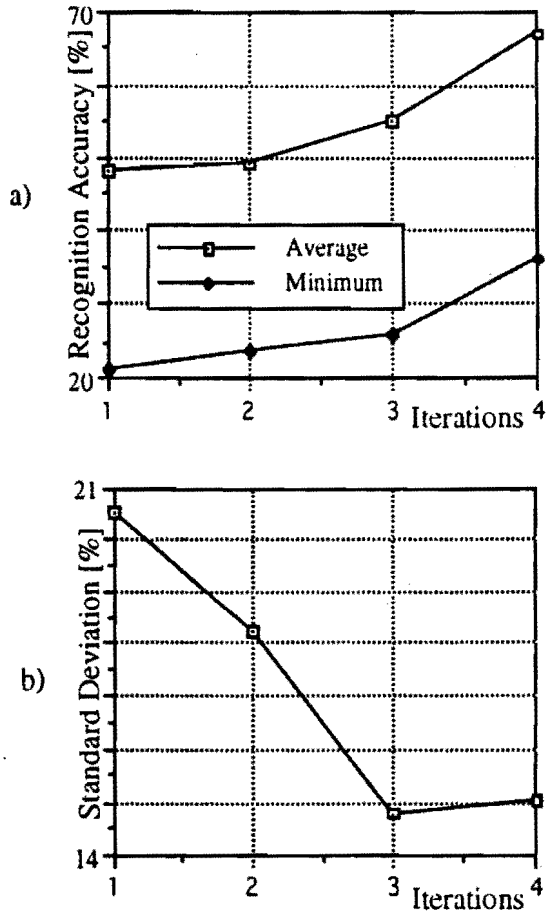


Figure 5-5: The twelve textures experiment

The obtained results are in accordance with the standard evaluation criteria for the recognition system that require: (i) an increase of the classification confidence when matching a class description with data belonging to this class, (ii) a decrease of the classification confidence when matching data with other class description, and (iii) perform on the similar confidence level for all classes when data is matched with their class descriptions. These criteria are expressed by (i) an average recognition rate computed through testing all twelve texture classes (we require the highest averaged recognition rate), (ii) a low deviation in the distribution of recognition rates from their mean value (this system stability criteria prefers the minimum value of standard deviation), and (iii) an improved recognition rate for the worst performing concept description (we seek the improvement of the minimum recognition rate, searched through all classes of texture considered in the learning process).

The most important conclusion drawn from these experiments is that the presented method is not greatly sensitive to the attribute extraction process selected. For a given extraction process, there might always be some texture class descriptions that cannot be used for the recognition because of incorrect classification results. The relevant/discriminatory information derived from these classes is not captured by the extraction process. Choosing other extraction process may help to generate better descriptions for these classes, but there might still be a different subset of texture set that performs poorly in the recognition phase. To alleviate this problem in the multilevel mode of the TEXTRAL method, correct/incorrect classification results are used as the essential class dependent information that help to discriminate between different classes by learning the next set of rules. This approach differs from traditional approach which tries to improve the effectiveness of recognition by designing more sophisticated extraction methods and which applies classifiers in such a way that they are adapted on the feature set to take optimal advantage of the extracted information. Such an approach belongs to the class of feature extraction oriented methods, where an extraction of relevant feature plays a very important role. The main problem with traditional approaches is the lack of a universal extraction method that works effectively with noisy data.

5.3 Experiments with Rule Optimization

This section presents experimental results with indirect optimization texture descriptions [Pachowicz and Bala, 1991]. These experiments were run for twelve texture classes (shown in Figure 4-1). The training data was extracted applying modified Laws' method with the radius of averaging window equal to 7.5 pixels. Training data for each class consisted of 200 examples. The test data was extracted from different areas of the same texture and consisted of 200 test events for each class.

5.3.1 Truncation of Less Significant Rules

The effectiveness of the simplest concept optimization method that incorporates the truncation of less significant concept components complexes was already investigated. [Zhang and Michalski, 1989] demonstrated the increase in the recognition effectiveness when this method was applied to simple domains of symbolic data. This data, however, did not match noisy vision domain.

When truncation of less significant concept components was applied to texture descriptions no increase in average recognition rate was observed. The recognition rate (for 12 classes) was equal to 73% for a wide range of optimization degrees; i.e., for removed complexes covering 2% to 30% of training examples. At the same time, the minimum recognition rate was constant and equal to 37%. The standard deviation applied to monitor system stability criterion oscillated randomly within a very small range of values; i.e., between 23. and 23.5. A simple truncation was inefficient when applied to optimize texture descriptions. However, the lack of a decrease in system performance suggests that the truncation of less significant concept components could be applied to reduce the size of concept descriptions without negative effects in the recognition effectiveness.

5.3.2 The SG-TRUNC Optimization Method

The AQ16 integrated learning system [Bergadano, Matwin et al., 1992] as a program based on the SG-TRUNC optimization method was used for these experiments..

The AQ16 program differ mostly from the AQ15 program in the implementation of the SG-TRUNC method and in the concept matching technique. The effectiveness of the AQ16 system was already demonstrated by [Zhang and Michalski, 1989] as an effective concept optimization method. The AQ16 integrated system, however, was tested with simple non-engineering data and has never been applied to texture data.

The first experiment with six textures was successful. The optimization increased the average recognition rate and decreased the deviation of the recognition rate. The data for the first experiment, however, was much simpler. Both training and testing data were less noisy, the attribute space was less complex, and number of classes was smaller to compare with 12 classes experiment.

For all texture data presented in Figure 4-1 (twelve classes), concept optimization performed by the AQ16 system gave poor results. These results are presented in Figure 5-6.

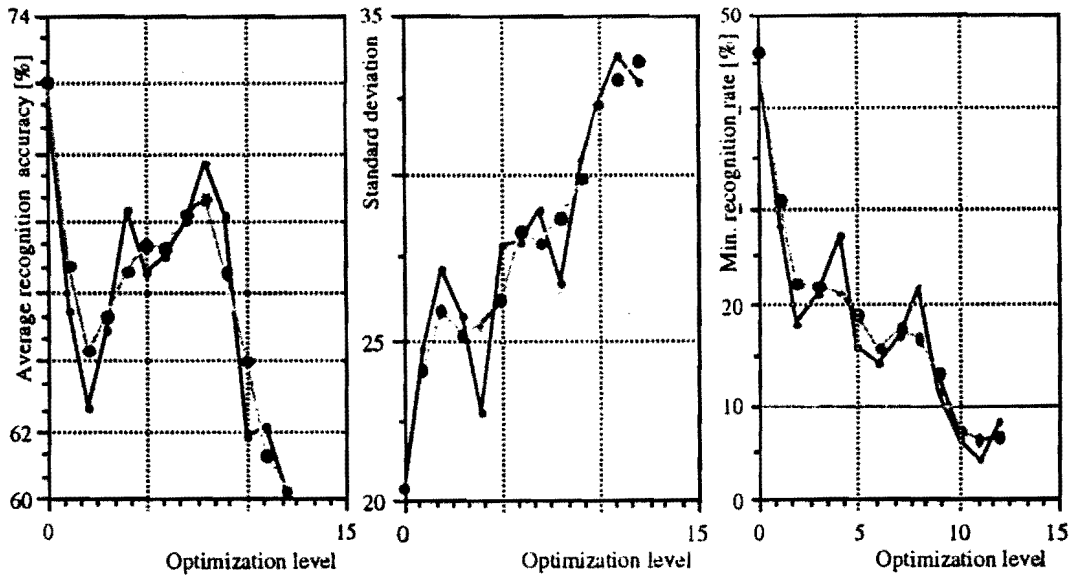


Figure 5-6: Recognition results for the SG-TRUNC optimization method

Solid lines in the diagrams represent average recognition rate, standard deviation, and minimum recognition rate when the values of optimization level were increased. Dotted lines represent smooth characteristics. The average recognition rate dropped from about 72% to 63% and then recovered slightly to the 70% level with the increase in optimization level. This recovery, however, was associated with a very fast increase in the value of standard deviation and with a rapid decrease in the minimum recognition rate. This means that the performance of well performing class descriptions was further increased while the performance of worse performing classes was decreased deeply. Obtained results indicated deep decrease in the stabilization of system recognition performance. At the same time the average recognition rate had no clear trend. These results motivated the research to seek other methods of concept optimization problem. The next chapter describes two newly developed methods.

Chapter 6

6. Other Concept Description Optimization Methods

In order to achieve completeness and consistency in the presence of noise, one may generate overly complex and detailed descriptions. Such descriptions, however, may not perform well in future cases and suffer the disadvantage of excessive complexity. This is the well known phenomenon called overfitting. In addition, there is another effect of noise on the performance of inductive learning. A disjunct (rule) of a concept description is usually formed by selecting a set of common properties shared by a group of positive examples and none of negative examples. These properties are selected on the basis of the distribution of positive and negative examples. When the level of noise is high, noise tends to confuse the selection of properties, because noise may change the distribution. This effect results in an incorrect description, even when inconsistency and incompleteness are allowed in the description.

Before advancing concept optimization methods, the following conclusions from the analysis of the optimization model (Section 4.6.1) can be stated [Pchowicz and Bala, 1991]:

- *Removal of less significant concept components of negative class descriptions does not imply automatic generalization of most significant components of a given class over the space released by removed components.*

- *The ultimate goal of such removal, however, should be the increase in typicality of concept components of a given class and the decrease in their number over the border areas between different concepts.*

- *Such an increase in typicality of concept components and the decrease in their number over the border areas between different concepts can be achieved indirectly; i.e., when pre-optimized concept descriptions are used to filtrate a final set of training data and the learning process is repeated with the new dataset.*

In order to derive homogenous areas representing concept descriptions and to improve borders between concept descriptions of different classes, one has to merge partitioned concept components. This merging can be executed correctly over the space released by the removal of less significant concept components if these components were incorrectly acquired as components of counterclass descriptions. Such generalization over released areas of attribute space (1) extends concept components describing main cluster areas; i.e., increases the typicality of these concept components, and (2) improves the separation of concept descriptions of different classes between themselves.

To implement such generalization over released areas of attribute space, an indirect optimization method was developed. Introduced method uses pre-optimized concept descriptions to exclude less significant concept components by the filtration of training dataset. If some primary training examples were noisy and thus produce less significant concept components then some of these noisy examples can be filtered by pre-optimized concept descriptions. It logically follows than that the filtered set of training data can be reused to learn final concept descriptions. The next section describes the AQ-NT method that uses pre-optimized concept descriptions to remove noisy training examples.

6.1 The AQ-NT Method

The AQ-NT method represents a novel way of handling the problem of learning from noisy real-world data [Pachowicz and Bala, 1991]. It is based on the idea that events covered by rules with a low t-weight may be noisy. The assumption is that the

system learning from a dataset that does not contain such events has a greater chance to produce correct (in the sense of predictive accuracy) concept descriptions than when learning from the original events.

The process of learning concept descriptions (in the form of a ruleset) is done in the following two phases (Figure 6-1):

Phase 1: Performs a rule-based "filtration" of the noise from the training data. This is done in the following way:

1. *Induce decision rules from a given dataset using the AQ learning program.*
2. *Truncate concept descriptions by removing "the least significant" rules, defined as rules that cover only a small portion of the training data (have small t-weight relative to the t-weight of other rules).*
3. *Create a new training dataset that includes only training examples covered by the modified concept descriptions.*
4. *If the size of the dataset falls below an assumed percentage of the training data (which reflects an assumed error rate in the data), then go to Phase 2. Otherwise, return to step 1.*

Phase 2: Acquire concept descriptions from the reduced training dataset using the AQ learning program.

The second step of the algorithm identifies a set of small disjuncts for each concept from the complete and consistent description generated in the first step. In the initial experiments small disjuncts are determined based on a threshold (TH) ranging from 0 to 1 provided by the user and the number of examples covered by the largest disjunct. The largest disjunct covers the most examples. If the number of examples covered by the largest disjunct is N, then all disjuncts which do not cover more than N * TH examples are small disjuncts. The method used to define small disjuncts is based on the number of examples covered by the largest disjunct rather than the number of total examples. Thus in the proposed algorithm, a concept description has small disjuncts only if it includes some large disjuncts. In the algorithm, the threshold TH controls the size of small disjuncts. Larger TH causes more small disjuncts to be selected and more examples may

be removed. When TH is set to 1, all disjuncts are small. When TH is set to 0, no disjunct is small. The TH should be set according to the degree of noise in a domain. Generally, in a highly noisy domain, TH should be set larger than in a less noisy domain. If TH is too large, many non-noisy example will be removed. If TH is small, the algorithm will do more iterations in which only a few examples are removed.

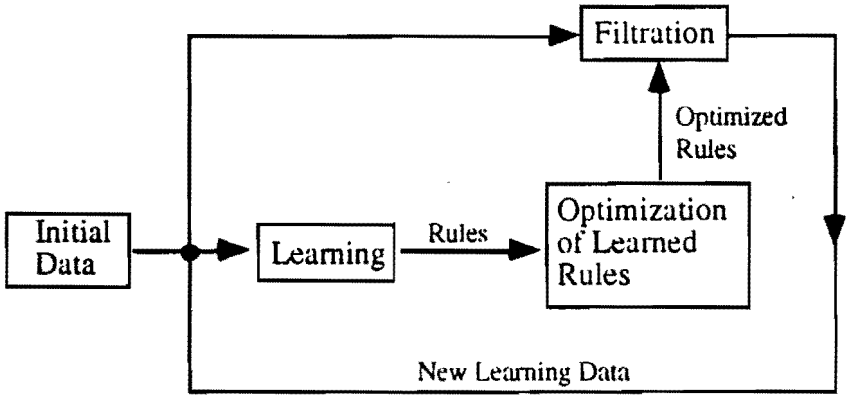


Figure 6-1: The AQ-NT flowchart

The proposed method has advanced rule truncation in two aspects. First, after negative noisy examples are removed from the training set disjuncts (rules) broken by the removed negative noisy examples could merge into one larger disjunct. In the rule truncation method, the remaining disjuncts never get a chance to be merged. Second, the AQ-NT algorithm removes noisy examples gradually, and each iteration removes only a subset of noisy examples that are easily identified. This prevents the removal of too many non-noisy examples. The descriptions generated from the initial training set may not include many large disjuncts, because of the noise. Many small disjuncts do not cover noisy examples. According to the measure we use to decide small disjuncts, only the disjuncts that cover one or two examples are chosen as small disjuncts. Therefore, only very few noisy examples are removed at the beginning. After some noisy examples are removed, some small disjuncts which cover non-noisy examples may merge into larger

disjuncts. Examples covered by some small disjuncts which do not merge are removed in the next iteration.

6.2 The AQ-GA Method

The AQ-GA is the second optimization method described in this chapter [Bala, DeJong et al., 1991], [Bala, DeJong et al., 1993]. This method integrates two forms of learning, symbolic inductive generalization and genetic algorithm based learning. The integration is done in a closed-loop fashion in order to achieve robust concept learning capabilities. The learning process cycles through two phases (Figure 6-2).

In the first phase, initial concept descriptions are acquired by running a noise-tolerant extension of the AQ15 rule induction system. The resulting concept descriptions may not be, optimal from the performance viewpoint, due to the AQ bias to generate simple, cognitively-oriented descriptions. Therefore, in the second phase, the system attempts to improve the performance of the descriptions by employing a genetic algorithm (GA).

The descriptions obtained from AQ15 are semi-randomly modified, using basic genetic operators: mutation and crossover. The resulting descriptions are evaluated according to a performance criterion. The criterion was the recognition accuracy of the descriptions on the "tuning" data (a subset of the training set of events). The best performing descriptions are selected from the population, and a new generation is repeated. The process stops when a desirable performance level is achieved, or the number of generations exceeds some limit.

Genetic algorithms typically represent individuals in a population (here, concept descriptions), using fixed-length binary strings. A novelty of the AQ-GA method is that it uses, instead of binary strings, concept descriptions (formally, VL_1 expressions) produced by AQ15. To this end, a special mutation operator was designed to introduce small changes to selected condition parts of the rules in each concept description. The condition parts are selected by randomly generating two pointers: the first selects a rule, and the second one selects a condition in this rule.

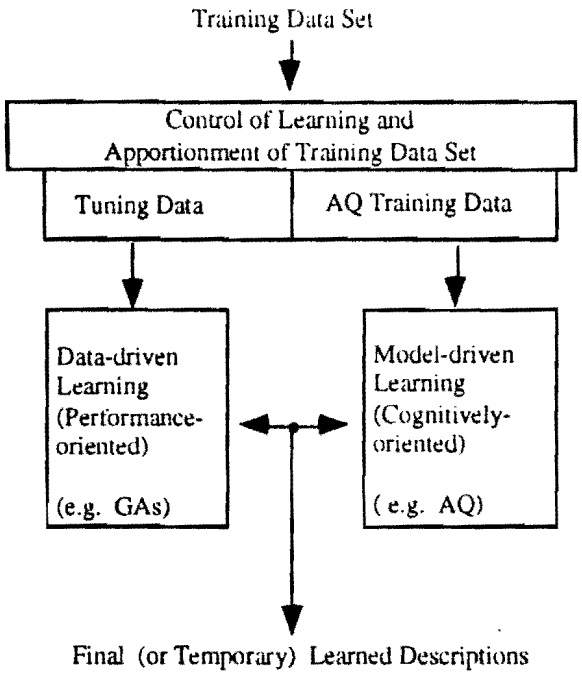


Figure 6-2: The AQ-GA method

The most-left or the most-right values of the referent in this condition are slightly modified. For example, the condition [x1= 10..23] might be mutated to any of the following: [x1 = 10..20], [x1 = 10..24], [x1 = 12..23] or [x1 = 8..23], as well as others. Such a mutation process samples the space of possible concept description boundaries to improve the performance criteria. The mutation process can be viewed as equivalent to various *transmutations* (knowledge transformations; Michalski, 1993) of the conditional part of a rule:

- specialization: [x5 = 3, 10..23] ==> [x5 = 3, 10..20]
- generalization: [x5 = 3, 10..23] ==> [x5 = 3, 10..24]
- variation: [x5 = 3, 10..23] ==> [x5 = 5, 10..23]

The crossover operation is performed by splitting concept description into two parts, upper rules and lower rules. These parts are exchanged between parent concept descriptions to produce new child concept descriptions. Since the degree of match of a given tuning event depends on the degree of match of this event to each rule of concept description, this exchange process enables inheritance of information about strong rules (strongly matching) in the individuals of the next evolved population. An example of crossover applied to short, four rules description is depicted below:

Parent description 1

1 [x1=7..8] [x2=8..19] [x3=8..13] [x5=4..54]

2 [x1=15..54] [x3=11..14] [x6=0..9] [x7=0..11]

----------crossover position ----------

3 [x1=9..18] [x3=16..21] [x4=9..10]

4 [x1=10..14] [x3=13..16] [x4=14..54]

Parent description 2

1 *[x1=16..54] [x5=0..6] [x7=5..12]*

2 *[x1=8..25] [x3=8..13][x4=9..11] [x5=0..3]*

----------crossover position ----------

3 *[x4=0..22] [x5=8..9] [x6=0..7] [x7=11..48]*

4 *[x2=5..8] [x3=7..8] [x4=8..11] [x5=0..3]*

The result of the crossover operation (one of two child descriptions) is the following:

1 [x1=7..8] [x2=8..19] [x3=8..13] [x5=4..54]

2 [x1=15..54] [x3=11..14] [x6=0..9] [x7=0..11]

3 *[x4=0..22] [x5=8..9] [x6=0..7] [x7=11..48]*

4 *[x2=5..8] [x3=7..8] [x4=8..11] [x5=0..3]*

6.3 Experiments with AQ-NT and AQ-GA

6.3.1 The AQ-NT Experiments

The AQ-NT method for advanced concept description optimization have been implemented and applied to texture data. These introductory experiments have been described in [Pachowicz and Bala, 1991, 1992]. Texture data used in the introductory experiments was composed of 6 texture classes (Figure 4-2).

The initial training set for learning included 200 training examples for each texture, and 200 test examples were used in the recognition process. Each example was represented by eight attributes and the value of each attribute was obtained using one of eight Laws' masks. A misclassification noise level of 30% was introduced to each class. n% misclassification noise level means n% of all training examples are noisy examples in the training set. n% of examples are randomly selected from all classes and the classes that they belong to are switched. The performance evaluation was based on two aspects, classification accuracy and description complexity. Classification accuracy was measured as the average percentage of correct classifications made by the concept description on all instances. Description complexity was measured by the average number of disjuncts involved in describing six classes. Figure 6-3 represents results obtained by the AQ-NT method. The best recognition rate (95.3%) was obtained for TH = 0.2 (94.33% -86.0% = 8.33 % increase of average recognition rate was obtained after 12 iterations). Figure 6-4 shows results obtained for a one-step noise removal, by truncating the small disjuncts according to the TH value. The best average recognition accuracy was obtained for TH = 0.75. and is 94.6% (slightly lower than the one obtained in the iterative approach).

Figure 6-5 shows the average number of disjuncts in the iterative approach. We can see that this number decreases from 37 to 3. In the one step truncation method the average number of disjuncts obtained for TH = 0.77 is 10. These numbers show that the iterative approach outperformed the one-step truncation method with regard to description complexity. These are the most encouraging results obtained by initial experiments with the iterative approach.

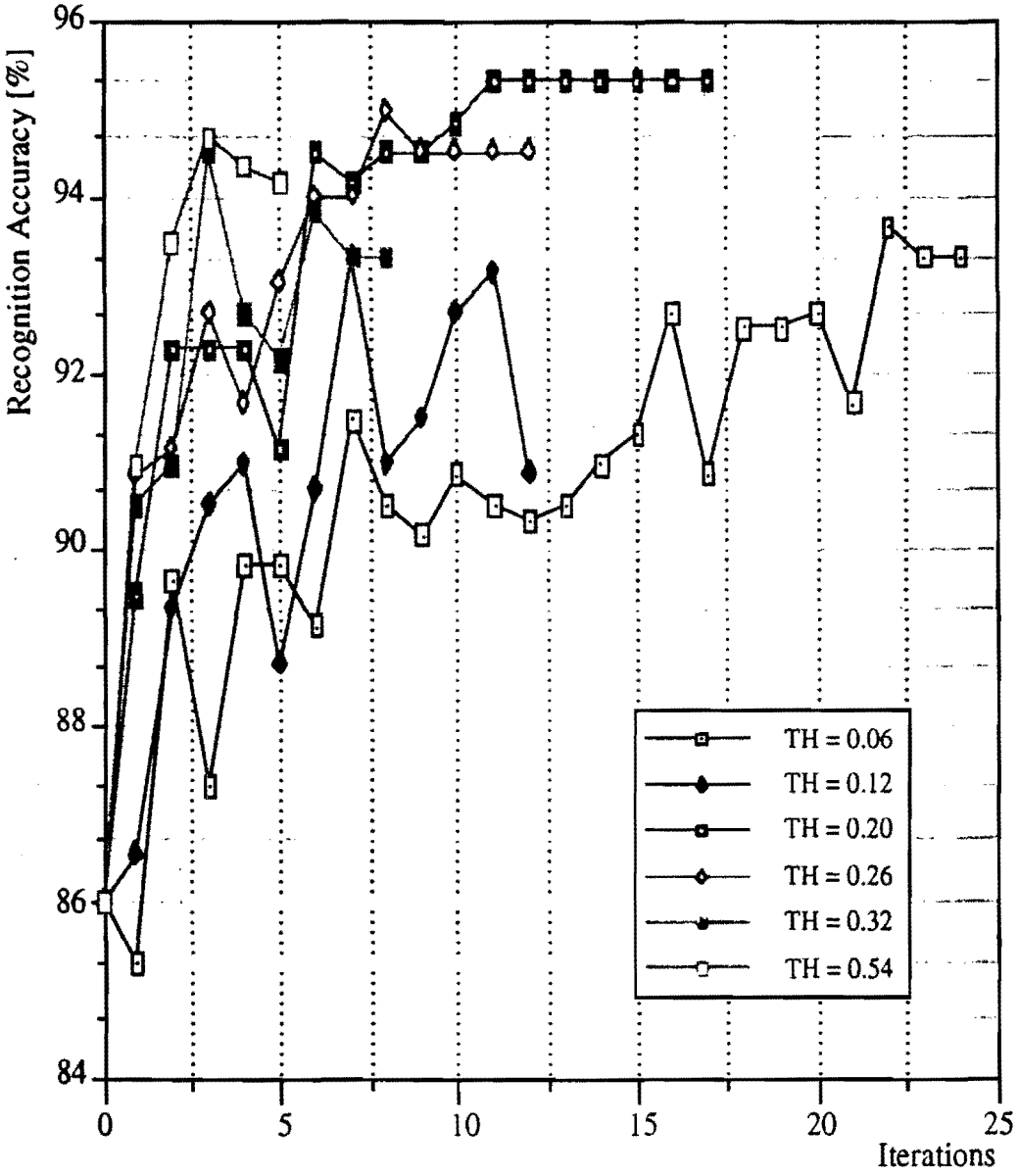


Figure 6-3: Average recognition accuracy

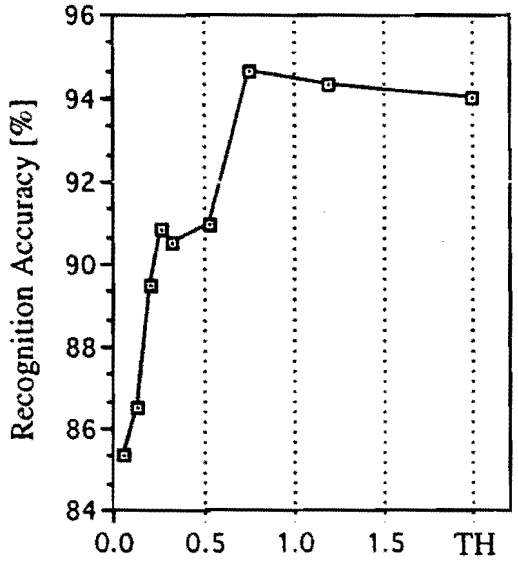


Figure 6-4: One step truncation (no relearning)

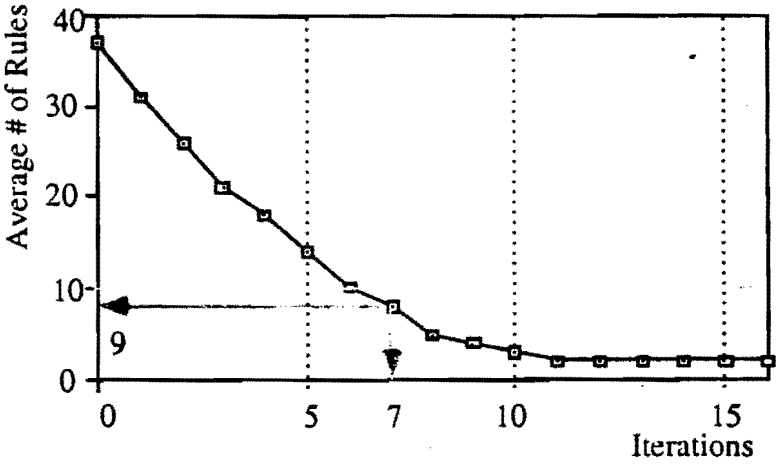


Figure 6-5: Average number of rules

The average number of disjuncts for one-step method (with TH=0.16) is 10 (dotted arrows in Figure 6-5). The iterative approach outperformed the one-step approach after seven iterations (9 rules).

Figure 6-6 depicts examples of two classes (A and F) distributed on the two attributes cross-section (x5-x6) of the representation space. One can observe the effectiveness of the AQ-NT method. Figure 6-6 (c) represents the situation after the AQ-NT run. Almost all noisy examples have been successfully truncated and the cross-section is similar to the cross-section without added noise.

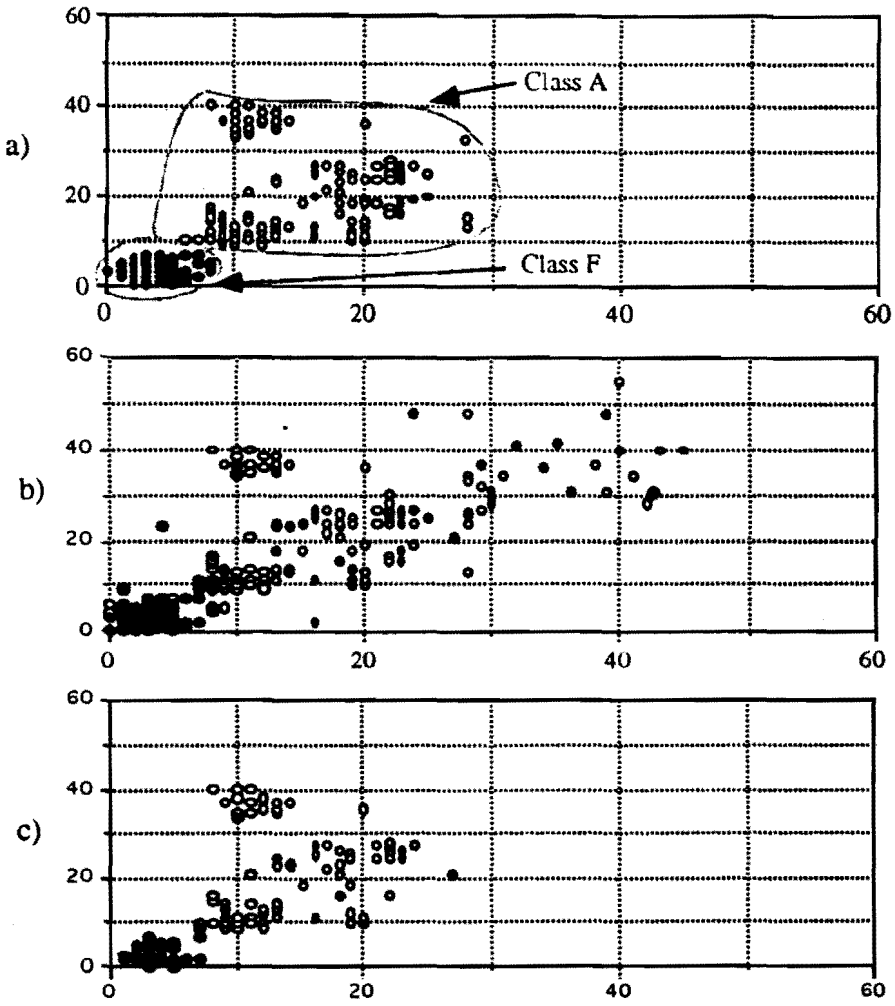
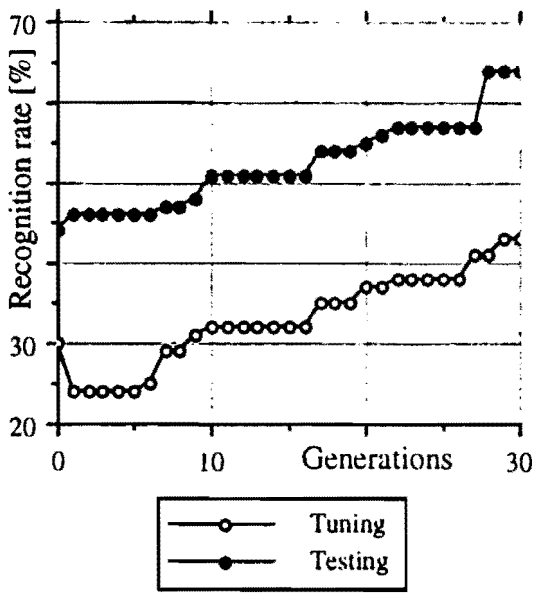


Figure 6-6: Cross-sections through attributes x5 and x6 of examples from class A (white dots) and class F (black dots): (a) clean noise-free data, (b) noise-added data, (c) data after the AQ-NT noise removal

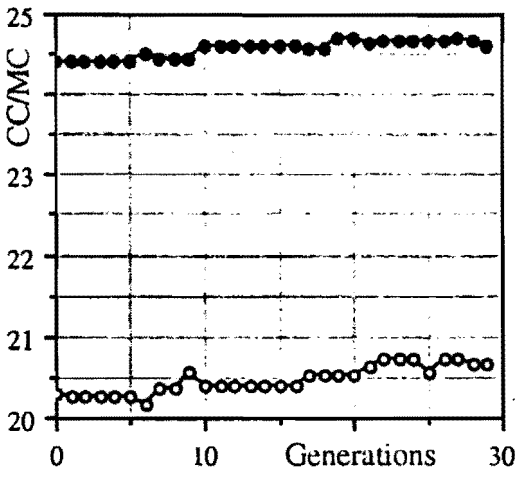
6.3.2 The AQ-GA Experiments

The performed experiments [Bala, DeJong et al., 1991], [Bala, DeJong et al., 1993], involving learning rules for describing texture classes, demonstrated that the classification results obtained with the hybrid learning algorithm (Figure 6-3) (AQ Training Data -> AQ and Tuning Data -> GAs) exceed the performance of the AQ algorithm used alone (AQ Training Data + Tuning Data -> AQ).

In the experiment we used 12 classes of texture data. The initial descriptions were generated by the AQ module using 200 examples per class. Another set of 100 examples was used to calculate the performance measure for the GAs cycle. The testing set had 200 examples extracted from different image areas other than training and tuning data. The "weakest" class (with the lowest matching results with testing data) selected for the experiment was class d92 (Figure 4-1) with 40 disjuncts (rules). Figure 6-7 represents results of the experiment. White circles of the diagrams represent characteristics obtained for tuning data used to guide the genetic search. Characteristics mapped by black circles were obtained for testing data. Figure 6-7 (a) shows the performance of genetically evolved description of d92 texture. When all 300 examples were used to generate rules, the average classification rate for this class (when tested with 200 examples) was below 45%. When the set of 300 examples was split into two parts, 200 for the initial inductive learning and 100 as the tuning data for GAs cycle, the correct classification rates obtained in the 30th evolution was above 60%. That is a significant increase in comparison with 45% obtained from inductive learning only. Figure 6-7 (b) represents the evaluation function used by genetic algorithm in order to guide the genetic search. The evaluation function was calculated as a rate of the correct classifications to mis-classifications for all twelve texture classes and is depicted for both testing and tuning data. The increases of CC/MC on both diagrams represent an overall improvement of system recognition performance. The system performance was investigated for a larger number of GA generation steps. However, it appears, that the noticeable increase was reached both for the d92 class description and for the overall system performance in a very few generation steps (i.e., in 10 steps).



(a)



(b)

Figure 6-7: Experimental results of the AQ-GA method

The effects on system performance due to running the GA for a larger number of generations was also investigated. However, it again appeared that most of the

improvement (both the decrease in complexity of the d92 class description and the increase in performance) was obtained in a very few number of generations.

6.3.3 The Segmentation Experiments

This section describes results of the application of the AQ-NT and AQ-GA system to texture segmentation problem. The following experiment was performed. An image 256 by 256 pixel positions with six areas of different textures was acquired --- Figure 4-2 with indicated classes: A, B, C, D, E, and F. A set of 200 examples was extracted to represent each texture class. A 50 % misclassification noise was introduced by shuffling examples between different classes. After introducing the misclassification noise, 150 examples in each class were examples from other classes. Such heavy misclassifcation noise was used to test how effectively obtained rules from these examples can be applied to texture segmentation task and how different noise removal techniques can improve image segmentation. Additional 200 tuning examples were extracted for each texture class. This set of examples was used by the AQ-GA method.

In the experiment, two methods were used to improve the rule performance. The first method was the AQ-NT method and the second, directly following, the AQ-GA method. In each iteration, obtained descriptions were used to segment the textural image. After 13 iterations of the AQ-NT algorithm the peak performance was obtained. Then, the worst performing concept description was identified and forwarded to the GAs module. Class F was the worst performing; i.e., the recognition rate was equal to 84%. In the GAs module, a population of 20 different variations of the rule description of class F was created. After 30 generations of GAs run the best version of class F description was plugged into the set of other class descriptions. The characteristics of the GAs run are presented in Figure 6-8 and 6-9. The recognition rate for the description of class F was improved significantly; i.e., from 84% to 90% over 30 generations of GAs. The effectiveness of applied GAs modification of rule description for class F is illustrated in Figure 6-10 on segmented images, where class descriptions were applied to segment the textural image during the AQ-NT run and after the AQ-GA run.

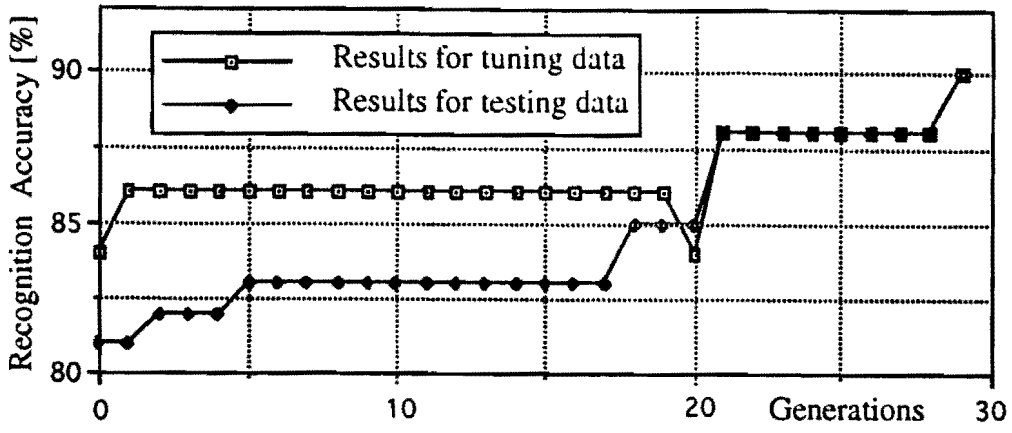


Figure 6-8: Recognition accuracy for the F class (AQ-GA run)

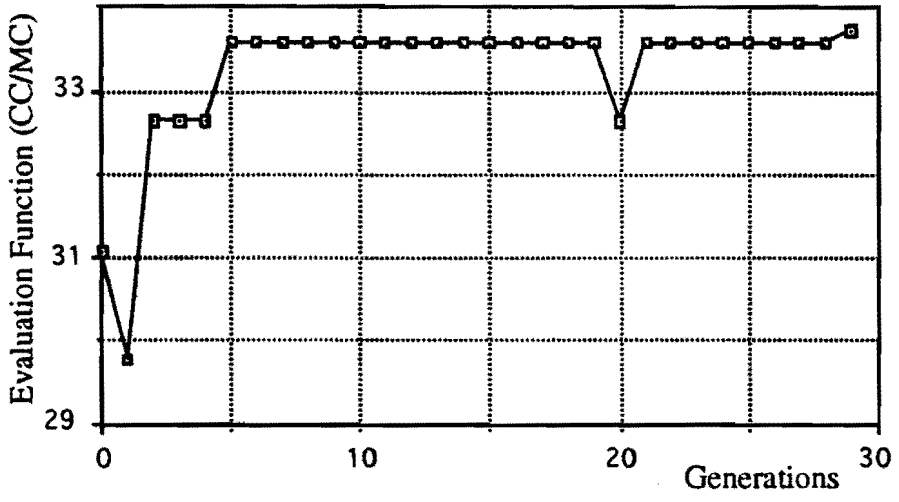


Figure 6-9: Evaluation function computed as the CC/MC for tuning data

The experiments with texture recognition and image segmentation showed that methods which use some heuristics can be used only to some point, where the only possible further increase of the performance can be accomplished by using some form of data-driven method. The best performance of the AQ-NT method was obtained after 13

iterations of the algorithm. The only further improvement of the performance was obtained by searching borderlines of condition subranges using a performance-based genetic algorithms.

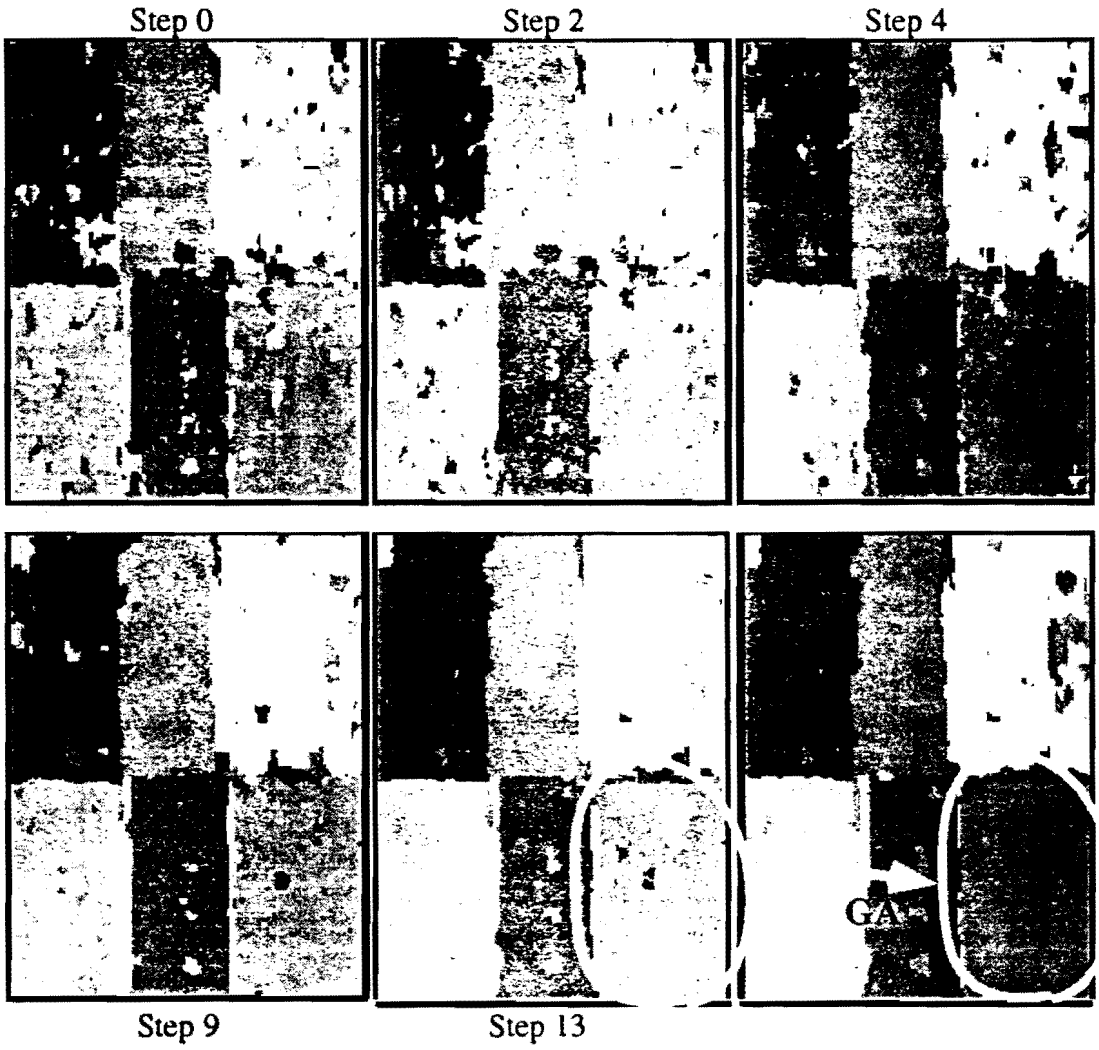


Figure 6-10: Results of the segmentation experiment with added 50 % of misclassification noise

Chapter 7

7. Related Work

7.1 Channic's TEXPERT System

TEXPERT is a software system designed by Channic [Channic, 1988] to acquire and apply knowledge for recognizing digital ultra-sound images. Originally, GEM (Generalization of Examples by Machine) [Reinke, 1984] was used as the learning mechanism in the system. Some experiments were then performed using a LISP version of AQ11 developed by Jeff Becker at the University of Illinois. Finally, TEXPERT used the AQ15 algorithm [Hong, 1986].

Experiments were performed on two-dimensional digital images obtained from ultrasound analysis of laminated aircraft materials. These images represent areas of varying thickness or structural flaws in the material. Figure 7-1 shows one of the digital ultra-sound images used in experiments.

In this image, there are two classes or areas. These are normal or class 0 (background) area and abnormal or class 1 area. In order to learn rules for classifying pixels into one of these two areas, the user selects a training area from which TEXPERT generates events to give as input to the learning algorithm. In most experiments Channic used a simple three-by-three pixel event template to generate twelve attributes for each pixel. These attributes consist of the intensity value of the pixel, the intensity value of the eight neighboring pixel, the maximum and minimum values among these nine, and the difference between the maximum and minimum value. The rules classified 94% of the class 0 (normal) pixels correctly and 97% of the class 1 (abnormal) pixels correctly.

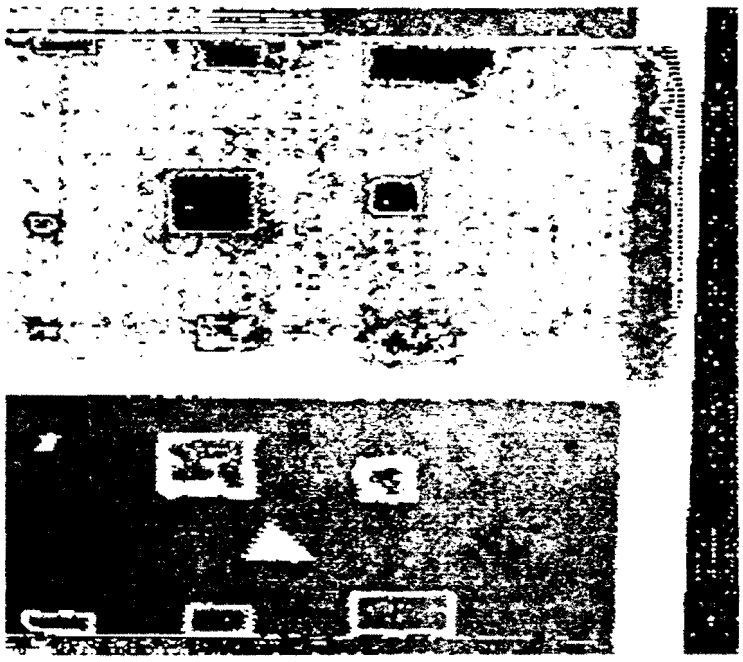


Figure 7-1: An example of ultra-sound image used by TEXPERT

7.2 Rule-based Versus K-NN Method

In [Pachowicz, 1989] a comparison of rule-based versus k-NN method is examined. The images used by Pachowicz were characterized by irregular lighting and several of them had smoothly changed resolution caused by the screw projection. The quality of images was low to create difficulties for the learning and recognition systems. Input gray-level images were processed by Laws' masks. A vector of k features (k=8) was extracted for a single pixel, and for each method of feature extraction. The scaling module performed the conversion of numeric features into their symbolic intervals. The consistency of learning data was checked, and in the case of inconsistent interval, the local scaling of higher resolution was performed. The AQ algorithm was applied to learn texture attributional descriptions from symbolic events. Next, these rules were

optimized incorporating two-tiered representation of imprecise concepts. Results are presented in Figure 7-2.

	PR approach k-NN	ML approach rules	ML approach optimal rules
Average recognition	**70%**	**80%**	**91%**
Highest recognition	99%	93%	98%
Lowest recognition	40%	72%	83%
Averaged deviation	**16.3**	**7.0**	**4.0**

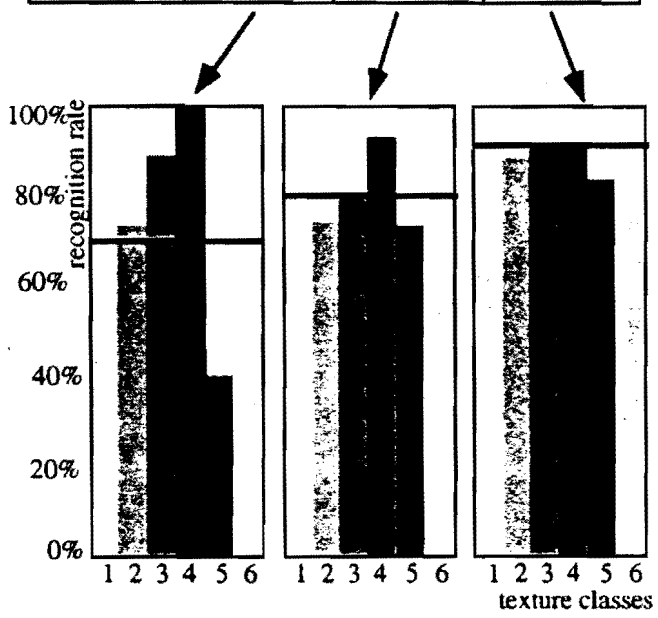


Figure 7-2: Comparison of machine learning and pattern recognition approaches

The best results (Figure 7-2) were obtained for the rule-based approach. The average recognition was equal to 91% and the system recognized all test textures correctly. The highest recognition rate was equal to 98% and the lowest recognition rate

was equal to 83%. The averaged deviation defined as $d = 1/N \sum |\underline{x} - xi|$ was the lowest and equal to 4.0 . The experiments that applied texture classes description using inferred non-optimized rules showed the average, highest and lowest recognition rates equal to 80%, 93% and 72%, respectively. The averaged deviation was greater and equal to 7.0. In comparison, the average, highest and lowest recognition rates for the k-NN (k=10) pattern recognition method were equal to 70%, 99% and 40%, respectively. In his work Pachowicz concluded:

1. The application of ML methodology increases the recognition effectiveness and optimizes the description of texture concept.

2. The improvement of recognition effectiveness possesses specific character; i.e., there is the increase of recognition rates for those classes that were classified before with lower rates, and the decrease of recognition rates for those classes that were recognized before with very high recognition rates.

3. Optimal rule description smoothes recognition rates; i.e., the deviation of the recognition rate is reduced.

In summary, the observed advantage of the applied ML approach is a four fold decrease of the deviation of the recognition rate. This effect gives the system more stability and makes the recognition more uniform.

7.3 Noise Tolerant Learning

Most of the early inductive learning systems, such as AQ11 [Michalski and Larson, 1978] and ID3 [Quinlan, 1979] make the "noise free domain" assumption that the examples presented to the systems do not contain any errors, and also assume that the description language is complete. These systems consequently constrain their searches to only those rules that are both consistent and complete. The requirement of "noise free domain" is often difficult to satisfy in engineering data [Chien, 1991], and therefore prevents many learning systems from being applied successfully to many practical problems.

There are two basic groups of approaches to learning from noisy data. One is to allow a certain degree of inconsistent classification of training examples so that the descriptions will be general enough to describe basic characteristics of a concept. This approach has been taken by the ID family of algorithms [Quinlan, 1986]. The main noise-handling mechanism for decision trees is tree pruning. There are two types of tree pruning: pre-pruning, performed during the construction of a decision tree, and post-pruning, used after the decision tree is constructed. The second approach is to discard some of the unimportant rules and retain those covering the largest number of examples. The remaining rules are a general description of the concept. Typical algorithms using these techniques are the AQ family of algorithms [Michalski, 1986]. Rule truncation in AQ15 ([Michalski, 1986, Zhang and Michalski, 1989, Pachowicz and Bala, 1991][Michalski, 1986, [Zhang and Michalski, 1989], and the significance test in CN2 [Clark and Niblett, 1989] are also examples of that approach. Other approaches are based on the minimum description length principle [Quinlan, 1989] and cross validation to control over-fitting during a training phase [Breiman, Friedman et al., 1984].

For example, an original technique has been developed and implemented within the CAQ program, a machine learning tool for engineering applications that integrates AQ and a statistical learning approach [Whitehall, Lu et al., 1990]. During the learning process, noisy events are removed during the process of building a consistent description by a method called distribution fit. The statistical approach in CAQ determines where the interval(s) for positive and negative events should be placed for a conditional part of a continuous valued attribute in the rule structure. An inconsistent conjunctive concept is constructed that still covers some negative events along with some positive events. The system needs to specialize the concept by adding a condition that will "uncover" the negative events and still leave covered as many positive events as possible. To reduce the amount of data that must be processed, the system looks at only the events currently covered by the conjunctive concept. It is in this section of the algorithm that events can be identified as noise. Noise events are removed from the process of building a consistent concept. The CAQ algorithm, however, assumes that a distribution of attributes is known a priori; i.e., the normal distribution.

7.4 The PRAX Method

In the TEXTRAL system, each texture class is represented by a ruleset. If there are very many texture classes, there will be correspondingly many rulesets, and the learning and recognition process may become complex. The PRAX method [Bala, Michalaki, Wnek, 1992] represents an alternative approach to the problem of learning a large number of concept descriptions.

The idea behind the PRAX method is to designate some concepts to be basic, and describe the remaining concepts in terms of the relations to the basic concepts. This idea can be simply illustrated by the example in Figure 7-3. If the system already knows the concept of "orange" (Des1) and "lemon" (Des2), then it can learn the concept of "grapefruit" by relating properties of the grapefruit to those of the lemon and the orange (Des3"), rather than in terms of original properties (Des3').

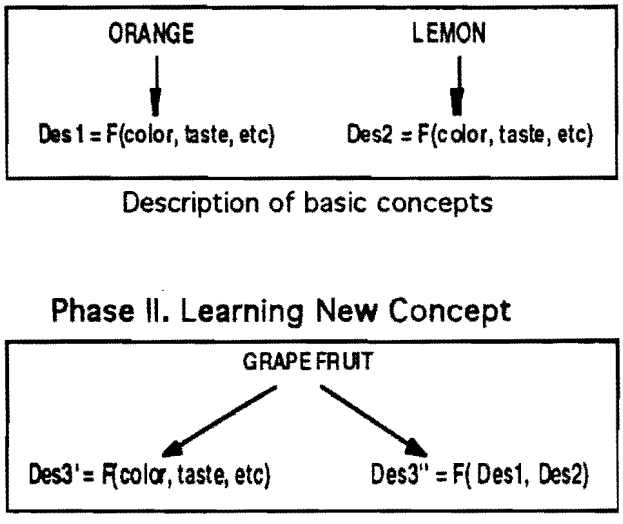


Figure 7-3: A simple illustration of the PRAX method

In the PRAX method, descriptions of the basic concepts are called "principal axes." They are learned in the similar way as in the TEXTRAL method. To learn a new, non-basic concept, the system determines a *similarity matrix* (SM) for that concept. The SM specifies the average degrees of similarity between the training examples of the new concept and all the principal axes.

The degree of similarity between an event and each principal axis is determined using flexible matching. The procedure determines the accumulated difference between the attribute values in the event and the conditions in each rule in the principal axis. To obtain a uniform representation of all class descriptions, the similarity matrix is also computed for all basic concepts.

These degrees of similarity can be viewed as values of the new constructed attributes. Thus, this method represents a special case of constructive induction. (The general concept of constructive induction includes any method that self-modifies the concept representation space during the induction process. Generating additional, problem oriented attributes is an important form of such self-modification of the representation space [Michalski, 1978; Wnek & Michalski, 1991].)

To recognize an unclassified event, the method creates an SM for it, that is, determines a matrix of similarities between the event and the principal axes. Subsequently, the system determines the best match between the SM of that event and SMs of all candidate concepts. The best match indicates the class membership.

The method was empirically evaluated by applying it to the problem of learning 24 texture classes from examples (Table 7-1). Each example was described in terms of eight multivalued attributes (representing detectors of various basic geometrical concepts, such as the presence of lines, edges, V-shapes, etc.). The performance of the PRAX-derived descriptions was compared with the performance of the k-NN classifier. Different level of misclassification noise were added to test the robustness of the method.

The main strength of the method lies in a problem-relevant transformation of the descriptor space. The new descriptors form generalized sub-spaces of the initial, training space. In addition, the method uses a non-linear distance metric to calculate values of

constructed attributes. The distance metric based on the idea of flexible matching is less sensitive to noise, then traditional Euclidean distance metric often used by pattern recognition methods.

Table 7-1: The results from comparing PRAX with the k-NN method

METHODS	The Recognition Rate (in %) of Examples from Unknown Texture
PRAX	
No Noise	100%
5% Noise	100%
10% Noise	100%
K-NN	
No Noise	96%
5% Noise	92%
10% Noise	87%

The current problem with the method is that it does not have a mechanism for deciding how to choose basic concepts. Choosing the minimal subset of concepts to be used for principal axes generation is important for method to be efficient. This problem will be a subject of future research. Another weakness is that the similarity matrix is a relatively complex representation.

Chapter 8

8. A Brief Summary and Conclusions

This dissertation research is a study of the applicability of machine learning methods to problems of computer vision. The first chapter discusses the background and motivations of the research. The specific problem investigated in this research is inductive learning of texture descriptions. The proposed schema for learning texture description is separated into; (i) image pre-processing and attribute extraction, (ii) acquisition of texture concepts by inductive learning, (iii) optimization of concept prototypes, and (iv) recognition of unknown texture samples.

In chapter 2, characteristics of sensory data and problems related to learning from such data are presented. Characteristics that can be expected in the vision data include: noise, complex representation spaces, large training sets, and large numbers of classes.

The general approach to learning textures is presented in chapter 3. Given an image with labeled samples of different textures (these samples are attribute vectors), the learning system generates sets of rules describing the different textures (Logical Templates). In an iterative mode of the learning method these rules are used to transform this image to a "symbolic" image, in which picture elements are labels of corresponding texture areas. New sets of rules (the next level template) can be learned from the symbolic image.

Chapter 4 is a detailed description of the learning method. A set of 12 textures is presented for experimental validation of the method. The set includes many random textures. This data results in attribute space with highly overlapping clusters of texture characteristics. The convolution method, operating on small image windows, invariant to changes in luminance and contrast, has been selected for extraction of texture characteristics. Inductive learning of texture concept descriptions was based on the AQ algorithm, developed by Michalski [1986]. The algorithm is a supervised covering algorithm. AQ learns decision rules by performing inductive inference on examples. Training examples are expressed as the conjunction of attribute values. The program performs a heuristic search through a space of logical expressions until it finds a decision rule that is satisfies all concept examples, but no counterexamples, and is optimized according to a preference criterion. Two classifications methods, the strict method and flexible method, were introduced to determine concept membership of an unknown example based on the descriptions generated.

Chapter 4 introduces a concept description optimization model and initial optimization methods are presented. One of the most important problems in the application of machine learning to vision data is the influence of noise. Image data is corrupted with noise, and consequently, learned concept descriptions contain noisy components. The main objective of optimization methods is to decrease the influence of noise on learned descriptions according to some performance measure. The following three performance measures have been proposed for texture recognition system: (i) increase the classification accuracy when matching a class description with examples of this class (*maximum accuracy criterion*), (ii) decrease the classification accuracy when matching examples with other class descriptions (*minimum misclassification criterion*), and (iii) perform on similar classification accuracy level for all classes when examples are matched with their class descriptions (*system stability criterion*).

To maximize the *accuracy criterion*, the learning method incorporates an iterative (multilevel) application of symbolic inductive learning to generate texture rules. In this mode, a learning process is repeated iteratively until the desired textural area transformation is obtained. The transformation of textural area is accomplished by matching examples extracted from all pixel position in the learning area with description

learned in the previous iteration. The result of this matching is a class membership decision obtained for each example. Since each example is extracted from some position in the learning texture area, this position is labeled with a class name.

The experimental results were reported in chapter 5. The main objective of all experiments was to show an improvement in performance (i.e., average recognition rate, standard deviation from the average rate, and minimum recognition rate). A significant increase in performance was achieved in the multilevel learning mode of the TEXTRAL method (12.5% increased of average recognition rate and 11% decrease of misclassification rate in the first experiment with multilevel learning). Experiments with the direct optimization of concept descriptions motivated the research to seek new optimization methods. In chapter 6, a novel indirect optimization method was proposed based on the analysis of a concept optimization model. The methods incorporates pre-optimized concept descriptions used to filtrate training data. The indirect optimization method outperformed direct methods. A method that uses genetic search was also introduced. The method is used to further improve the performance of the worst performing descriptions, and at the same time improve the whole set of concept descriptions. This methods significantly improved the system stability criterion. A surprising result for the Genetic Algorithms approach was that the method showed an improvement in just a few generations. The experimental results showed support for the proposed inductive learning from vision data.

Chapter 7 reports on related work. Channic's TEXPERT method is presented [Channic, 1988] The TEXTRAL method presents a significant advancement over the TEXPERT method. The improvement of TEXTRAL over TEXPERT lies in its application of inductive learning to fine-grained, high resolution random textures, utilization of concept optimization methods, and in its application of inductive learning to a larger set of textures.

Chapter 7 presents related work by [Pachowicz, 1989] on a comparison of rule-based versus k-NN method applied to texture recognition. The main advantage of the observed advantage of the rule-based approach is a four fold decrease of the deviation of

the recognition rate. This effect gives the system more stability and makes the recognition more uniform.

Finally, related noise tolerant learning methods presented in chapter 7 use different approaches. One is to allow a certain degree of inconsistent classification of training examples so that the descriptions will be general enough to describe basic characteristics of a concept (e.g., in decision trees a pre-pruning, performed during the construction of a decision tree, and a post-pruning, used after the decision tree is constructed). The second approach is to discard some of the unimportant rules and retain those covering the largest number of examples. Because the above methods try to remove noise in one step, they share a common problem — the final descriptions are based on the initial noisy data. For methods applying pre-pruning, the attributes used to split the instance space in a pruned tree are selected from initial noisy training data. For methods applying pre-truncation, the search for the best disjunct is also influenced by noise in training data. This problem is more severe for post-pruning and post-truncation. The post learning concept optimization cannot merge broken (by noisy examples) concept components (i.e., disjuncts, subtrees). So, the "gaps" in concept descriptions remain unfilled. Post learning optimization will cause the complexity of concept descriptions to decrease only if the concept components are eliminated; i.e., without reorganizing concept descriptions.

8.1 Contributions

Texture analysis has been an active research area in computer vision for more than two decades, and has proved to be a very difficult problem. This difficulty largely stems from the diversity of natural and artificial textures, which makes a universal definition of texture impossible. This dissertation focuses on a particular approach to texture analysis which is referred as the *rule-based* approach. The method generates symbolic descriptions of texture by first, deriving structural features from texture, and then generating covers (descriptions called Multilevel Logical Templates) to fit texture data. It can be viewed as the form of model-based approach to texture analysis (Section 1.1.2). Compared to other approaches, the rule-based approach is more general and applies to a large number of textures (successful experiments have been performed with both random

and structural textures). This generality is a direct consequence of reliance on the automatic generation of models by the inductive learning process. Learned models are optimized to increase their descriptiveness for the texture concepts they represent.

It has been stated that the primary goal of this introductory research is to explore inductive learning approaches to computer vision. Accordingly, the major contribution of this work is a demonstration that symbolic learning methods can be successfully applied to selected problems of low-level vision, in which nonsymbolic methods have been traditionally employed. To demonstrate this, three approaches were respectively implemented: TEXTRAL, the primary method for learning texture descriptions, AQ-NT for learning of reduced complexity rule sets from noisy inputs, and AQ-GA for rule enhancement. More specific contributions of these methods are summarized below.

TEXTRAL uses a novel representation of visual concepts. Visual concepts (texture class as in this thesis) are expressed in the VL1 (Variable-valued Logic System 1) [Michalski, 1972]. This representation was used for the first time in learning from fine-grained random visual textures. VL1 consists of the decision rules expressed as symbolic descriptions involving relations among object attributes. Such descriptions can enhance image understanding capabilities of computational vision systems by being more expressive, conceptualized and meaningful with the regard to visual object that they represents. Such description can be intelligently manipulated to enhance their performance and utilization within the vision system. This thesis describes some methods that manipulate the structure of the description in order to achieve better performance. The use of rule-based descriptions of visual concepts can also be utilized by constructive induction learning method. Such methods self-modify the concept representation space during the induction process. One important form of self-modification is generation of new attributes. Generating new problem oriented attributes of visual concept can address the problem of initial attributes selection and can definitely enhanced performance of recognition systems. A constructive induction method for texture recognition was address in the PRAX algorithm [Bala, Michalski et al., 1992]. In experimental testing of the method on the problem of learning descriptions of 24 visual textures, the PRAX method significantly outperformed the k-NN classifier.

A novel accomplishment of the research is the use of an iterative learning method to generate rules from fine-grained, noisy texture data. In this method, the learning process is repeated iteratively until the desired textural area transformation is obtained. The transformation of textural area is accomplished by matching examples extracted from all pixel position in the learning area with the description learned in the previous iteration. The results of this matching is a new symbolic image of the textural training area. The novelty of this approach is that correct/incorrect classification results obtained from one level and represented by the symbolic image are used as the class dependent characteristics. These characteristics are input to the learning algorithm to generate next set of rules.

The contribution and novelty of the AQ-NT approach is that the noise detection in training data is performed on the higher level of concept description rather than by analyzing training data. The AQ-NT method can be viewed as the model-driven data filtration. This resulted in a more effective methods than traditional data filtration applied to the raw data level only. The second novel aspect is that noisy examples are gradually removed by permanently searching for the least significant concept components as candidates for their removal. This gradual process enables the learning algorithm to reformulate new, better performing rules.

Genetic algorithms typically represent individuals in a population, using fixed-length binary strings. A novelty of the AQ-GA method is that it uses, instead of binary strings, concepts descriptions (formally VL1 expressions). Special mutation and crossover operators were designed to introduce semi-random changes to the rule structures. The mutation process can be viewed as equivalent to various *transmutations* (knowledge transformations; Michalski, 1993) of the conditional part of a rule.

The important contribution of the AQ-GA method is that it represents a new class of multistrategy learning approaches. The resulting concept description generated by inductive learning system may not be, optimal from the performance point of view, due to bias of inductive learning techniques to generate simple, cognitively-oriented descriptions. Therefore, inductive learning strategy is enhanced by performance-oriented

genetic search strategy. Thus, the method integrates two learning strategies — inductive and genetic. Although there exists substantial differences between symbolic induction methods and genetic algorithms (learning algorithm, performance elements, knowledge representation), their integration as presented in this thesis serves as a promising example that a better understanding of abilities of each approach can lead to novel and useful ways of combining them.

The texture data was the initial domain of experiments presented in this thesis. However, the methods can be applied to learning in any domain characterized by continuous attributes, noisy data, multiclass environments, and complex representation spaces. Characteristics of such a domain are common for engineering data. Learning from engineering data require new learning tools that are noise-tolerant, capable of processing complex data including multiclass environments, and large training sets. These new tools are expected to significantly extend the state of the art and open up whole new application areas for machine learning.

8.2 Limitations of Methods and Presentation

There are several limitations of the presented methods and presentation. They are listed below:

(i) Convolution operators were used in the presented experiments. They are invariant to small changes in luminance and contrast. However, the sensitivity of the methods to significant changes in environment were not addressed. This is the primary limitation of the current implementations.

(ii) It is unclear how the performance of the basic method depends on a very large number of texture classes. Recently, a modification of the basic method for the incremental learning in a multiclass environment has been reported [Bala, Michalski, and Wnek, 1992] (the PRAX method).

(iii) The question of efficiency for real-time applications was not directly addressed. Future research methods will aim at introducing parallel computation techniques to the methods (e.g., parallel utilization of a rule set during the recognition phase)

(iv) The user must define a set of attributes for the TEXTRAL system. Although the basic method has been tested with different attributes [Bala, 1990], it is not quite evident how the choice of attribute sets affects the performance. Constructive induction methods are expected to alleviate this problem to some extent.

(v) The initial comparison between rule-based and nearest neighbor approaches has been performed (Section 7.2). It reported better performance and stability of rule-based approach. However, it is yet not clear how the presented method compares to a neural-network approach.

(vi) To take the advantage of the genetic search in the AQ-GA method, it is required that a concept description has some minimum number of rules. Each rule represents a building block of a individual in population If the number of the building blocks is low the genetic search will have too few sampling points and may not converge. This limits the AQ-GA method applicability to highly disjunctive descriptions.

(vii) In the AQ-NT method the TH parameter (used to decide on a candidate rule for truncation) must be set up by the user. In the algorithm, the threshold TH controls the size of small disjuncts. Larger TH causes more small disjuncts to be selected and more examples may be removed. When TH is set to 1, all disjuncts are small. When TH is set to 0, no disjunct is small. The TH should be set according to the degree of noise in a domain. This is a limitation of the current implementation of the AQ-NT method.

(viii) In the evaluation of match between an unknown example and a rule a flexible matching method is used. It has not been established how other matching methods (based on different distance metrics) may perform.

8.3 Future Research

There are several other major topics to be investigated in future research:

(i) Enhancements to the current learning methodology to include capabilities for automatically generating higher level problem-relevant attributes (constructive induction).

(ii) Encoding higher-level knowledge of the image scene (e.g., combining texture with shape of regions or boundaries to enhance the recognition capabilities)

(iii) Learning in a "differential attribute space". The main idea is that the system must learn two descriptions; one that captures the learned concept itself, and the other that expresses changes in the concept due to environmental changes. These two descriptions can be used to achieve robust recognition capabilities.

(iv) Development of a method for dynamic recognition based on the texture models stored in the memory.

(v) The applicability of multistrategy learning (e.g., combining symbolic rule learning with neural network learning; the issue of representing and learning of imprecisely defined visual concepts).

(vi) Extensions of the methodology to other problems in vision, e.g., learning of shape classes.

(vii) Learning new visual concepts in terms of differences and similarities form known concepts, and developing a calculus for representing symbolic differences between visual concepts.

REFERENCES

Aloimonos, Y. and M. Swain, "Shape from Patterns: Regularization," *International Journal of Computer Vision*, Vol. **2**, pp. 171-187, 1988.

Bajcsy, R. and L. Lieberman, "Texture Gradient as a Depth Cue," *Computer Graphics and Image Processing*, Vol. **5**, pp. 52-67, 1976.

Bala, J. and J. Pachowicz, "Application of Symbolic Machine Learning to the Recognition of Texture Concepts," *Application of Symbolic Machine Learning to the Recognition of Texture Concepts*, Miami Beach, FL, 1991.

Bala, J. and R. Michalski, "Recognition of Textural Concepts Through Multilevel Symbolic Transformations," *The 3rd International Conference on Tools for Artificial Intelligence*, San Jose C.A., 1991.

Bala, J., "Combining Structural and Statistical Features in a Machine Learning Techniques for Texture Classification," *The 3rd International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, IEA/AIE-90*, Charleston, SC, 1990.

Bala, J., K. DeJong and P. Pachowicz, "Integrated Inductive Learning And Genetic Algorithms For Texture Recognition," *Workshop on Integrated Learning in Real-World Domains*, Aberdeen, Scotland, 1992.

Bala, J., K. DeJong and P. Pachowicz, "Using Genetic Algorithms to Improve the Performance of Classification Rules Produced by Symbolic Inductive Methods," *The 6th International Symposium on Methodologies of Intelligence System*, Charlotte, NC, 1991.

Bala, J., K. DeJong and P. Pachowicz, *Multistrategy Learning From Engineering Data by Integrating Inductive Generalization and Genetic Algorithms*, Machine Learning, Kaufman, Morgan, San Mateo CA, 1993.

Bala, J., R. Michalski and J. Wnek, "The Principal Axes Method for Constructive Induction," *The Ninth International Conference on Machine Learning*, Aberdeen, Scotland, 1992.

Ballard, D. and C. Brown, *Computer Vision*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1982.

Bergadano, F., S. Matwin, R. S. Michalski and J. Zhang, "Learning Two-tiered Descriptions of Flexible Concepts: The POSEIDON System," *Machine Learning*, Vol. **8**, pp. 5-43, 1992.

Breiman, L., J. H. Friedman, R. A. Olshen and C. J. Stone, *Classification and Regression Trees*, Wadsworth, Belmont, 1984.

Brodatz, P., *Textures: A Photographic Album for Artists and Designers*, New York, 1966.

Channic, T., "TEXPERT: An Application of Machine Learning to Texture Recognition," *M.S. Thesis*, University of Illinois, Urbana-Champaign, 1988.

Chien, S., Whitehall, B., Dietterich, T., Doyle, R., Falkenhainer, B., Garrett, J., Lu, S., "Machine Learning In Engineering Automation," *Proceedings of the Eight International Workshop on Machine Learning*, Evanston, Ill., 1991.

Clark, P. and T. Niblett, "The CN2 Induction Algorithm," *Machine Learning*, Vol. **3**, pp. 261-284, 1989.

Cross, G. and A. Jain, "Markov Random Field Texture Models," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. **PAMI-5**, pp. 149-163, 1983.

Davis, L., M. Clearman and J. Aggarwal, "An Empirical Evaluation of Generalized Cooccurrence Matrices," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. **PAMI-3**, pp. 214-221, 1981.

DeJong, K., "Learning with Genetic Algorithms: An Overview," Machine Learning vol. 3; 123-138, Kluwer Academic Publishers, 1988.

Derin, H. and H. Elliott, "Modeling and Segmentation of Noisy and Textured Images Using Gibbs Random Fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. **PAMI-9**, no. 1, pp. 39-55, 1987.

Devijer, P. and J. Kittler, *Pattern Recognition: A Statistical Approach*, Prentice Hall, 1982.

Du Buf, J., M. Kardan and M. Spann, "Texture Feature Performance for Image Segmentation," *Pattern Recognition*, Vol. **23**, no. 3-4, pp. 291-309, 1990.

Duda, O. and P. Hart, *Pattern Classification and Scene Analysis*, John Wiley & Sons, 1973.

Fisher, D. H., and Schlimmer, J.C., "Concept Simplification and Prediction Accuracy," *Proceedings of the 5th International Conference on Machine Learning*, Ann Arbor, MI, 1988.

Haralick, R., K. Shanmugan and I. Dinstein, "Texture Features for Image Classification," *IEEE Transactions on Systems, Man and Cybernetics*, Vol. **SMC-3**, pp. 610-621, 1973.

Hintz, K. and W. Bugert, "Set Normalized Density Dimension for IR Image Charcterization," *Submitted to IEEE Trans. on Image Processing*, Vol. 1991.

Holte, R. C., Acker, L.E., and Porter, B.W., "Concept Learning and the Problem of Small Disjuncts," *Proceedings of IJCAI-89*, Detroit, MI, 1989.

Hong, J., Mozetic, I., and Michalski, R. S., "AQ15: Incremental Learning of Attribute-Based Descriptions from Examples, the Method and User's Guide," *Report ISG 86-5,*

UIUCDCS-F-86-949, Computer Science Dept., University of Illinois at Urbana-Champaign, 1986.

Hsiao, J. and A. Sawchuk, "Supervised Textured Image Segmentation Using Feature Smoothing and Probabilistic Relaxation Techniques," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. **11**, no. 12, pp. 1279-1292, 1989.

Iba, W., Wogulis, J., and Langley, P.W., "Trading Off Simplicity and Coverage in Incremental Concept Learning," *Proceedings of the 5th International Conference on Machine Learning*, Ann Arbor, MI, 1988.

Laws, K., "Textured Image Segmentation," *Ph.D. Thesis*, University of Southern California, Department of Electrical Engineering, 1980.

Lu, S. and K. Fu, "A Syntactic Approach to Texture Analysis," *Computer Graphics and Image Processing*, Vol. **7**, no. 3, pp. 303-330, 1978.

Markovitch, S. and P. D. Scott, "The Role of Forgetting in Learning," *Proceedings of the 5th International Conference on Machine Learning*, Ann Arbor, MI, 1988.

Michalski, R. S. and J. B. Larson, "Selection of the Most Representative Training Examples and Incremental Generation of VL1 Hypotheses: The Underlying Methodology and the Description of Programs ESEL and AQ11," *Technical Report*, 867, Computer Svience Dept., University of Illinois, 1978.

Michalski, R. S., "A Theory and Methodology of Inductive Learning," *Artificial Intelligence*, Vol. **20**, pp. 111-116, 1983.

Michalski, R. S., "A Variable-Valued Logic System as Applied to Picture Description and Recognition," in *Graphic Languages* , Vol. Nake, F. and A. Rosenfeld, North Holland, Amsterdam, pp. 1972.

Michalski, R. S., "AQVAL/1 - Computer Implementation of a Variable-Valued Logic System VL1 and Examples of its Application to Pattern Recognition," *Proceedings of the First International Joint Conference on Pattern Recognition*, Washington, D.C., 1973.

Michalski, R. S., "How to Learn Imprecise Concepts: A Method for Employing a Two-Tiered Knowledge Representation in Learning," *Proceedings of the 4th International Machine Learning Workshop*, University of California, Irvine, 1987.

Michalski, R. S., "Pattern Recognition as Knowledge-Guided Computer Induction," *Technical Report*, 927, Computer Science Dept., University of Illinois, Urbana, 1978.

Michalski, R. S., Mozetic, I., Hong, J., and Lavrac, N., "The Multi-Purpose Incremental Learning System AQ15 and its Testing Application to Three Medical Domains," *Proceedings of AAAI-86*, Philadelphia, PA, 1986.

Michalski, R., J. Bala and P. Pachowicz, "GMU RESEARCH ON LEARNING IN VISION: Initial Results," *1993 DARPA Image Understanding Workshop*, Washington, DC, 1993.

MLV-92, "Materials on the NSF/DARPA Workshop on Learning and Vision," (forthcomming report), 1992.

Pachowicz, P. and J. Bala, "Texture Recognition Through Machine Learning and Concept Optimization," P91-12, MLI 6, Center for Artificial Intelligence, George Mason University, 1991.

Pachowicz, P. W., "Low-level Numerical Characteristics and Inductive Learning Methodology in Texture Recognition," *Proceedings of IEEE International Workshop on Tools for Artificial Intelligence*, Fairafax, VA, 1989.

Pchowicz, P., "Learning Invariant Texture Characteristics in Dynamic Environments," MLI-2-91, Center for Artificial Intelligence, George Mason University, 1991.

Pentalnd, A., "Fractal-based Descriptions," *IJCAI*, Munich, Germany, 1983.

Poggio, T. and C. Koch, "Ill-possed Problems in Early Vision: From Computational Theory to Analog Network," *Poc. Roy. Soc. (London)*, Vol. 1985.

Pratt, W., O. Faugeras and A. Gagalowicz, "Visual Discrimination of Stochastic Texture Fields," *IEEE Transactions on Systems, Man and Cybernetics*, Vol. **SMC-8,** pp. 796-804, 1978.

Quinlan, J. R., "Induction over Large Data Bases," *Heuristic Programming Project*, HPP-79-14, Computer Science Dept., Stanford University, 1979.

Quinlan, J. R., "Learning Relations: Comparison of a Symbolic and a Connectionist Approach," Basser Dept. of Computer Science, University of Sydney, Australia, 1989.

Quinlan, J. R., "Simplifying Decision Trees," *International Journal of Man-Machine Studies*, Vol. **27,** pp. 221-234, 1987.

Quinlan, J. R., and Rivest, R.L., "Inferring Decision Trees Using the Minimum Description Length Principle," *Information and Computation*, Vol. **80,** no. 3, 1989.

Quinlan, J. R., *The Effect of Noise on Concept Learning*, Machine Learning: An Artificial Intelligence Approach, Vol. II, R.S. Michalski, J.G. Carbonell and T.M. Mitchell (Eds.), Morgan Kaufmann, Los Altos, CA, 1986.

Reinke, R., "Knowledge Acquisition and Refinement Tools for the ADVICE META-EXPERT," UIUCDCS-F-84-921, Department of Computer Science, University of Illinois, 1984.

Rendell, L., "A New Basis for State-space Learning Systems and A Successful Implementation," *Artificial Intelligence*, Vol. **20,** 1983.

Rosenfeld, A. and B. Lipkin, "Texture Synthesis," in *Processing and Psychopictorics* , Vol. Academic Press, New York, pp. 717-728, 1970.

Rosenfeld, A., J. Aloimonos and L. Davis, "Maryland Progress in Image Understanding," *DARPA Image Understanding Workshop*, 1990.

Tambe, M., and Newell, A., "Some Chunks Are Expensive," *Proceedings of the 5th International Conference on Machine Learning*, Ann Arbor, MI, 1988.

Tcheng, D., Lambert, B., Lu, S.C-Y, and Rendell, L., "Building Robust Learning Systems by Combining Induction and Optimization," *Proceedings of IJCAI-89*, Detroit, MI, 1989.

Tomita, F. and S. Tsuji, "Extraction of Multiple Regions by Smoothing in Selected Neighborhoods," *IEEE Transaction on Systems, Man and Cybernetics*, Vol. **SMC7,** pp. 107-109, 1977.

Unser, M. and M. Eden, "Multiresolution Feature Extraction and Selection for Texture Segmentation," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. **11,** pp. 717-728, 1989.

Vistens, R., "Texture Models and Image Measures for Texture Discrimination," *International Journal of Computer Vision*, Vol. **3,** pp. 313-336, 1989.

Weiss, S. M. and N. Indurkhya, "Reduced Complexity Rule Induction," *Proceedings of IJCAI-91*, Sydney, Australia, 1991.

Whitehall, B., S. Lu and R. Step, "CAQ: A Machine Learning Tool for Engineering"," *International Journal for Artificial Intelligence in Engineering*, Vol. **5,** no. 4, 1990.

Wnek, J. and R. Michalski, "Hypothesis-driven Constructive Induction in AQ17," *IJCAI-91 Workshop on Evaluating and Changing Representations in Machine Learning*, Morik, K, F Bergadano and W Buntine, Sydney, Australia, 1991.

Zhang, J. and R. S. Michalski, "A Description of Preference Criterion in Constructive Learning: A Discussion of Basic Issues," *Proceedings of the 6th International Workshop on Machine Learning*, Segre, A., Cornell University, Ithaca, NY, 1989.

Zhang, J. and R. S. Michalski, "Rule Optimization via SG-TRUNC Method," *Proceedings of EWSL-89*, Montpellier, France, 1989.

Zucker, S., A. Rosenfeld and L. Davis, "Picture Segmentation by Texture Discrimination," *IEEE Transactions on Computing*, Vol. **C-24,** pp. 1228-1233, 1975.